April 1988 Vol. 1 Nº 7 Price £1.50

Archive

The Subscription Magazine for Archimedes Users

Mandelbrot and Julia Patterns

Using FORTRAN 77

Using the Debugger

Which ROM Podule?

Which Multi-sync Monitor?

Using the I/O Podule

BBC to Archimedes Program Conversion

Variable/procedure cross-referencer for BASIC V

Bouquets or Brickbats?

Thank you so much to all those people who have written and said how much they appreciate what we are trying to do here at Archive magazine. It makes such a difference to know that at least someone appreciates what you are doing!

That leads me on to think how the folk at Acorn must feel with all the brickbats we seem to be slinging at them. Yes, I know they deserve a lot of them, but do we ever tell them in our letters and calls that we do appreciate what they are doing? After all, they have given us a superb computer to play with (sorry, make use of); they are trying hard to recover from problems like the RS423 fiasco, which, to an extent, wasn't really their fault anyway; and they are trying to provide a good back-up service within the constraints put on them (from Olivetti, presumably).

How about an Easter Resolution: Be nice to an Acorn employee!

Hail the Conqueror!

My review copy of Conqueror arrived the other day. My first reaction, when I saw that it was all about tank warfare was that I did not like the idea at all, especially in the light of the Hungerford massacre and similar atrocities. Why do so many computer games involve the shoot 'em up mentallity?

Under pressure from the kids, I tried it out. As a piece of simulation software, it really is **very** impressive, including sampled sounds, courtesy of Armadillo. It is also more than just a straight us versus them battle; it involves quite a bit of strategic thinking.

Then I realised that if I reviewed it in this issue of Archive and got Conqueror in stock, being the first such review available, I could sell quite a large number of programs at £20+. A very attractive proposition!

Then I had a word with the Boss about it – He didn't say "No!" (He's good like that.) but I still felt a bit uneasy about the whole thing, so I decided in the end not to stock it after all.

Archive

Volume 1 • Nº 7 • April 1988

Contents

Hardware & Software Available

• **Artisan Support Disc.** Clares' have just released a support disc for Artisan users. Amongst other things, it provides 6 extra fonts, extra colour fill patterns, an editor for changing the colour patterns on the monochrome printer dumps, printer dumps for Integrex 132 and Epson EX800, JX80, Fujitsu DL3400 and Star LC10 colour printers and a module which provides a way of merging pictures (including a fastload module) – you can make up a very impressive Artisan "slide-show" – and you get four new sample screens to impress your friends. (In stock and available through Archive for £18) The Integrex screendumps are very impressive as visitors to the Manchester M.U. Show will be able to testify, especially when printed on the special paper which Clares Micros can supply for £5 per 25m roll.

• **New version (1.5a) of Arctist** available – still at £19.95 (£18 through Archive). Now has the following extra features: Select colour & tint from palette or an area on-screen (i.e. re-select an existing colour), finish a line or ray by pressing space then restart elsewhere without re-selecting the function, select a new colour while still in a function, fill a given colour area with a new tint level, block move straight or EOR overlap, clear screen to a selected colour, delete text from the keyboard, make use of twin drives, rotate through palettes, select a new palette.

This new version is available as a FREE UPGRADE to existing users (Acorn, please note!). There are also Econet and hard-disc versions available – telephone Fairhurst Instruments for details.

• **5.25" disc drive Interface.** All you ever wanted to do with a 5.25" disc drive on the Archimedes from CJE Micro's... Firstly an interface cable for £30. Then there is a link-swapping switch for £8 that will allow the external drive to be :0 (or, more importantly, A: on MS.DOS). Then there's a Drive Select Translation Cable for an extra £9 which allows you to leave the links on the external drive set to zero (so you can swap the drive over with another computer, e.g. a BBC) and still have it accessed from the Archimedes as if it were drives 1 and 2 (or there is a version that makes it 2 and 3). If you can't work out what you need from that lot, give CJE a ring! Then there is some software for £15 that allows you to read (but not write) DFS discs using the Archimedes – much easier than trying to use the RS423 interface – particularly useful for those who want to trade in their BBC's for an Archimedes! (All these prices are including VAT)

• **Art-Worker** (£5) from MacSoft is a "creative art program" providing a spray brush for creating mode 15 (256 colour) pictures.

• **Mode Converter + Serial Data Transfer System.** (£13 + VAT) from M & M Computers provides a way of getting all those favourite screen-shots (or other files for that matter) across from the BBC to the Archimedes. Once across, the mode converter will allow you to transfer screens from one mode to another so that, for example, 16 colour BBC pictures can have a bit more colour added. (M & M are moving into new offices, so for information, please contact the author, Alan Barclay at 7 Porthill Court, Aberdeen, AB1 1DU.)

• **Architext** – a text editor from HopeSoft for use with C Pascal, Fortran, assembler and BASIC programming – £19.90. We have a review copy though it is "not the final version" but you can get a demo version which allows you to do everything except save files for just £3.50 (refundable when you buy the full program).

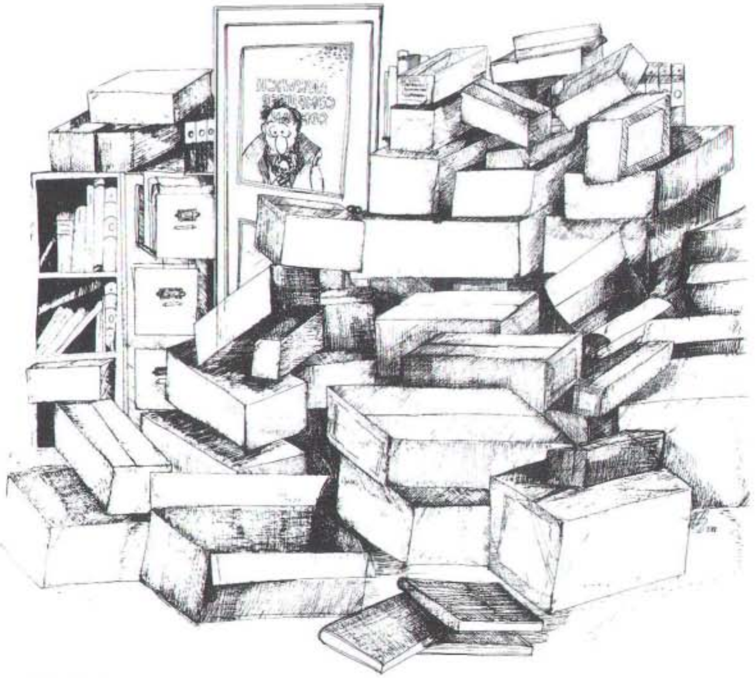
Architext uses the mouse but not the WIMP environment to make it "fast and full of facilities, but easy to use.")

• **Backgammon on your desktop** – Maximum have released a desk accessory version of the popular board game, backgammon with save, load, wind and re-wind facilities for just £9.50. You can play the computer or the computer can plays itself!

• **AutoSketch** (£79 + VAT) – a new precision drawing package for machines with 1Mbyte upwards is object-oriented with lines, arcs, circles and curves of different colours and line types which can be assigned to any of ten drawing layers. Text of any size can be added and objects can be measured and dimensioned automatically. It has full zoom facilities, is mouse driven and works with a whole range of printers and pen-plotters. You can add pictures to your own symbol library and then paste them into other drawings. For more details, contact Michael Page at Acorn (Fulbourn Road).

• **New Adventure Game** – Bush Rescue £21 +VAT from 4Mation Educational Resources. An educational adventure program set in Australia – very appropriate in Bicentennial year!

• **What a bind!** Archive PVC binders (£5.50 each) are now in stock. I'm very pleased with them – they are strongly made and look very smart on the bookshelf. The binding system is such that even with only six or seven magazines they stay in place without any trouble.



"If you order 500 binders at a time", he said, "they work out a lot cheaper." I worked out the cash flow OK, but I didn't think about the storage! They arrived today in thirty huge boxes! So if you're thinking of buying one "sometime", can I urge you to buy it now? Please!??

Comment Column

No more Technical Help from Acorn?

If you try to ring Acorn's previously very helpful Customer Services Department, they will tell you that they are unable to give technical advice over the phone and advise you to talk to your local Acorn Dealer. The idea is that if your Acorn dealer doesn't know the answer he can get through on a special hot-line to Acorn, try and explain your problem to them and then relay the message back to you. (Shades of "Send three and fourpence, we're going to a dance"!)

Ask SID!

Dealers (and you too, if you want to subscribe – £40 p.a.) can also ask SID – Acorn's own Support and Information Database – a round-the-clock viewdata standard bulletin board. It has three main sections: a set of pages of "continuously up-dated general and technical advice", information on new products and a bulletin board facility allowing you to communicate with each other and with SID's editorial team.

Look out SID, here comes CHARLIE!

We are looking at the possibility of setting up a bulletin board specifically for the Archimedes. (SID supports all Acorn products). This would probably be free to all Archive subscribers. It would allow you to pose your technical queries; you could send in your hints and tips (and even longer articles, subject to consultation with the editor); you could download the Archive program listings (though I think we would make some charge for this); we could put Archimedes public domain software onto it etc, etc. If you have any views or good ideas or just want to encourage us to go ahead with the project, please get in touch.

(There's a prize for anyone who can work out what CHARLIE stands for!)

No more free Arc-Writers?

A quote from a letter from Acorn to one of our readers: "Registered owners before 31st March will receive a free word processor for the Archimedes. Alternatively, we will be releasing First Word Plus next month (March) with built-in spelling checker and the facility to include graphics in the text". I presume they don't mean that First Word Plus is to be the new freebee so I gather that they are no longer going to be giving away a word-processor with the Archimedes.

Another Emulator Up-grade?

No doubt you have paid your £15 for the up-grade PC-Emulator disc (version 1.09). Well now there's a rumour that there is an up-graded MS-DOS disc for a further £10! It will "...automatically increase the MS-DOS application workspace to around 540k on a 1Mbyte machine."

(Actually, that report was from one reader, but when another reader called in at Fulbourn Road, they said they hadn't heard of any MS-DOS up-grade! Any ideas anyone?)

The A410 – Worth the wait?

The latest information I have from Acorn is that the A410's with the newly designed printed circuit boards will not be available until the end of July. I suspect that Acorn may not hurry that project along too much because as soon as the 410 becomes available, people will be able to up-grade the memory and add a cheaper hard-disc and end up with the equivalent of a 440 for somewhat less money.

If this is the case, it seems that Acorn are cutting their own throats. They may be able to persuade a few people that it is not worth waiting for the 410 and get them to buy a 440, but by and large all they are doing is putting more and more people off the whole project. Acorn have produced a superb machine that has terrific potential and now they are jeopardising the whole thing by not producing the goods as and when promised. Did I hear you say something about "déjà vu"? Well, to be fair, the BBC micro did eventually take off in a big way, but I wonder if that was as much due to influences from outside Acorn than from within.

RS423 Yet Again

From reports we've been getting, the RS423 fix from Acorn for the GTE 6551 chip seems to fix the problem of **receiving** at 9600 and 19200, but it seems to have introduced a problem when **transmitting** at those speeds.

Still, there is hope!!! I've heard that one chip manufacturer is about to launch a version of the chip which **will** work on the Archimedes. What I don't know, however, is what Acorn's policy will be about up-grading. I think that the machine as it stands is "not fit for the purpose for which it was sold" and therefore we will be within our rights to ask Acorn to fit the new chips retrospectively as a field change through dealers, but we'll just have to wait and see.

(STOP PRESS – New version of GTE fix – 1.24 – just arrived – not tested but will put on program disc.)

(The changes that are needed to eliminate the buzz are now apparently recognised by Acorn as field changes, so your local dealer should be able to do the job for you.) A

Matters Arising...

• **Fast screenload/save**. Due to a 'feature' of the 1.2 operating system you will find that the text cursor may appear on screen after a screenload. To avoid this, add line 1935 SWINE "XOS_RestoreCursors".

• **Backplanes** – It seems that the backplane for the 400 series works OK on the 300 series machines. In other words there is no reason why you should not put four 8-bit podules on the 300 machines. You cannot, however, put any 32-bit podules on a 300-series machine (if there were any 32-bit podules available!!!) because the extra address lines are not available on the p.c.b. connector.

• The **1Mbit chips** that you would need to put into a 305/310 to increase its memory to 4M are 18 pin devices whereas the sockets are only 16 pin, so the answer to the Fact or Fantasy statement on this issue is Fantasy! So says one reader, but another says that there is a pin compatible chip and that the operating system (which is the same as is in the 400 series) can detect which it has and accesses them accordingly. Solidisk apparently managed to get 4 Mbytes going on 300 series machines but it involved cutting tracks in the interleaving layers of the p.c.b. by drilling a hole in the board.

• **Hard Disks.** To clear up the speculation over the hard disk interface on the Archimedes, it is not SCSI but the Shughart ST506 standard.

• **Stopping the drive light coming on** when accessing a 5.25" disc attached to a Viglen drive – remove links marked "FG" and IU".

• **Slower listings.** I was rightly chastised by one reader for saying that non-BBC users would probably not know about holding <ctrl> and <shift> down together to halt the listings. On the contrary, non-BBC users would have read the

(Matters Arising continued...)

manual, rather than thinking they knew it all, and discovered a reference to it on page 16 (even referred to in the index under "listings, halting") and would also have discovered that holding down <ctrl> on its own slows down the listing to a steady, readable (well, viewable) speed. So now it's one finger on <ctrl> and the other on and off <shift> for me.

• Sorry about the misprint on the order form last month. The **Integrex colour dump** program is **£25** as it says in the text, not £20. A

Hints and Tips...

• One reader says he has solved the problem of **"To linefeed or not to linefeed..."** by adding a single pole switch in parallel with the contact on the printer dip-switch that sets the auto-linefeed on and off, leaving the dip-switch in the open position, of course. Don't forget though that most printers only look at the dip-switch settings at power-up, though some do so when you do a software reset with 27,64.

• With **Arc-Writer**, to get it **not** to double space on a Panasonic printer, use the AW_PCedit program, select the RX80D and change the number of steps per 100 units vertically from 21600 to 600.

• **View 3** works under the emulator if you poke locations A8A1, A8A2 and A8A3 with &EA.

• Acornsoft **ISO Pascal, Comal, Prolog and LISP** all work under the emulator says one reader.

• **Share Analyser** (Synergy Software) works perfectly under the 6502 emulator.

• **Four floppies on 1.2 OS.** If you try *con. floppies 4, you will find that it tells you the number is too big, but if you use *FX162,135,4 and do a <ctrl-break>, you will finds that *STATUS tells you that you have four floppies.

• **Booting other drives.** In contrast with the BBC micros, you can, on Archimedes, *MOUNT 1 and then <shift-break> to boot the disc in drive 1.

• **WIMP Manager.** Be aware that the first time Wimp_poll is called, it clears the screen.

• **Fitting the backplane:** It is an awkward job to fit the backplane because two plastic spacers are used into which self-tapping screws have to be inserted. This is best done in a vice to avoid stripping screw heads and/or sticking screwdrivers through your fingers!

• **'Unknown IRQ at &00000000'.** If you get this error message (and possibly find that the machine hangs up on you) check whether your RS423 lead is properly screened, if at all. If it is not screened, it seems to be able to pick up interference (remember the micro-wave problems mentioned in issue 5, page 8?) and this generates spurious interrupts.

• More ideas on **the black art of RS423 communication!** Here is one reader's recipe for succesful RS423 transfer, though it may not work for everyone... Try using the 'fix' module (Archive 1.6 program disc) with pins 1,4,8 and 9 linked together and 6 (DSR) and 7 (RTS) as the control lines. New fix on disc 1.7!

• **Running more than one BASIC program.** If you have a BASIC program you want to run while you are working on another program in memory, you can program a couple of keys with:

```
*KEY 4 P%=PAGE : PAGE=&30000
        : CHAIN "testprog"|M
*KEY 5 PAGE=P% : OLD|M
```

• **System Delta Plus:** If you have trouble loading newly formed databases and get "No Room" when you know there is plenty of room, add a .D to the filename when loading, i.e. type in the filename as, say, "RECORDS.D".

• **Inter-Word:** If you have come from the desktop, you will find that the tab key does not work. This can be corrected with *FX219,9.

If you want the £ sign, you will find it is produced by the ` key (far top left key).

If you want the ` sign... tough! (The £ sign duplicates the "insert ruler" function!)

• **Stuck in a black hole?** If you find yourself typing in black on black, try typing MODE MODE<return>. It allows you to type again, but maintains the mode you were in.

• **SystemDevs module.** The redirection commands, some of which we said didn't seem to work, are sensitive to spacing. The correct forms are as follows where the spaces are represented by bullets (•):

```
*CAT•{•>•info•}
```

Spools catalogue to file "info"

```
*CAT•{•>>•info•}
```

Adds catalogue to file "info"

```
*BASIC•{•<•data•}•PROG
```

Runs "PROG" and uses information in file "data" for input.

• **Using the Acorn ROM Podule:** No links are available for the RAM sockets, although you can configure the sockets by software for 8k, 32k or 128k RAM chips. Using HM6264LP-15 ram in the sockets and configuring them as 8k does not immediately work. However if you configure them as 32k chip and apply an offset of &2000 when loading i.e.:

```
*RMLoad <podule number>
      <socket> <filename> 2000
```

then your data will load correctly. This is because the Chip Select (CS) on the 8k chips is on the pin Address 13 (pin 26) on the 32k and 128k. This causes the 8k chip to be mapped into 8-16k and 24-32k hence the offset of 8k. The inverted CS on pin 20 is common to all three devices. (Without configuring them as 32k, the operating system reports that the device is not big enough for an 8k offset.)

• **Concerning Acornsoft C and TWIN**: 'Concurrency' (keeping text in one window and compiling a program in the other) does not work. The compiler (v1.5A) has a bug in it that prevents it from working. Acorn said that it used to work OK on the pre-release version! However, compiler output can be sent to a file called tmp.temp if the -spool option is used.

It may be necessary to relocate TWIN in order to compile programs. My version of TWIN loaded at &60000 but needed to be relocated to &80000 to allow the compiler enough space. (See below.) If the compiler crashes, try increasing the SystemSize using *configure.

The two compiler messages, 'Fatal I/O error' and 'Binary output error', simply mean that your disc is full!

• **More about TWIN.** Twin is an editor which seems to have one or two hidden features; firstly it is a relocatable program, and consequently can be loaded anywhere where RAM exists (barring overwriting OS workspace etc.). I have created a version which I call LoTwin which loads at &10000; this leaves space for BASIC programs only 256 bytes fewer than are available on a BBC Master using Shadow RAM, and yet it allows me considerably more work space than the standard Twin.

To create a version of Twin called 'MYTWIN' which loads and runs other than at the 'normal' address, perform the following actions:

```
*DIR <dir'ry_containing _Twin>
*LOAD TWIN <new_load_address>
*EX
*SAVE MYTWIN <new_load_
                address> +<size>
```

<size> is obtained from the 3rd column of hex data after the '*EX'.

Then, to invoke the new version, type

```
*MYTWIN
```

and it will run. Pressing <shift-f5> followed by <T> after entering MYTWIN will show the load address on the top line immediately to the left of the time.I have yet to see mentioned in print the 'warm start' for TWIN. If something nasty happens and your Archimedes locks up just as you have spent a couple of hours typing in something using TWIN, and you have no option but to press <ctrl-break>, or even <ctrl-reset>, do not despair, simply type:

```
*GO <twin_start_address> -WARM
```

and you will find yourself back in TWIN exactly as you were when the 'nasty' happened, barring any corruption caused by the 'nasty'. Don't forget the '-' preceding the "WARM'! I have found myself in this position after a momentary power cut, and although the warm start did not restore TWIN as a fully functional editor (due to corruption), it restored Twin's ability to save text, so I saved what I had done, and was then able to reload Twin, and then reload my text.

• **Iso Pascal:** Be warned before you buy Iso Pascal, it has **NO** support for Arthur – otherwise, says one reader, it is an excellent implementation. Also (in common with C) it comes with a very thin manual, so a copy of TWIN and a book about the language (see the mini-review on page 25) will be needed for most people.

(More information about other languages was sent in, but this was all I could digest with my limited knowledge of languages other than BASIC. Would anyone be prepared to receive all the "other languages" information and knock it up into a special section each month?)

Using the Computer Concepts ROM Podule

How many times have you had to press <ctrl-break> and muttered under your breath because you've lost your function key definitions? That need never happen again if you have Computer Concepts's ROM podule with some battery-backed RAM on it. All you do is configure the system to start up in the RFS instead of the ADFS and also configure it to auto-boot. Then in the RFS, you put a boot file something like:

```
REM > RFS:!BOOT
*SET Run$Path ,ADFS:$.,RFS:$.%.
*ADFS
*FX255,8
PRINT "Acorn ADFS"'
*KEY 0 These are
*KEY 1 my favourite
*KEY 2 key definitions
*SET ALIAS$> Cat
*SET ALIAS$? HELP etc
*BASIC
```

(You will notice that the Run$Path uses '$' instead of '%' as stated on page 28 of the Computer Concepts manual.)

• To get the **Inter series software** to boot up with a simple call such as *ISHEET, copy 65Arthur into the RAM area using *COPY ADFS:Modules.65Arthur RFS:65Arthur and, providing you have set up the Run$Path as above, *ISHEET, *IWORD and *ICHART will automatically load the emulator and then the appropriate software.

• **Installing Wordwise Plus**. If you copy the Archimedes version of Wordwise Plus from Computer Concepts's disc (filename "WW+") into the RFS calling it, say, WWIMAGE, you can modify the BASIC program "ISHEET" and save it as "WW+" so that, as above, you can just type *WW+ and it will auto-load the emulator and run the software. The modifications are to change the references to "ISIMAGE" at lines 130 and 270 into "WWIMAGE".

• If you want **to modify any of the programs in a ROM**, all you do is either LOAD it, edit it and SAVE it or, if you have the modified version on disc, just *COPY it. If you tell it to SAVE a file with a name that exists in ROM it gives the ram version precedence over the ROM version.

To find out whether a particular file is in ROM or RAM, use the *INFO <filename> command. If it comes up with a "*" before the final figure of the information, the file is in the RAM area. To get the ROM version back, you either have to do a <ctrl-break> or type *RMREINIT RFS <return> because *delete filename removes all trace of it so that even the ROM version is inaccessible.

Thanks to Adrian Look for all the following hints and tips.

• **X-SWI's.** When using SWI's if you put an X before the name then the SYS call from BASIC will not generate an error. For example:

```
    SYS "XOS_CLI",block
```

• **Modes & drawing speed.** The VIDC receives the video data down the data bus lines of the ARM processor. While this is happening the RISC chip cannot perform any processing. This means that the screen modes using higher memory will be slower. Try timing a FOR-NEXT loop of 1,000,000! So, if speed is of the essence, it may be better to go into mode 0, do the calculations and save them in an array and then go into mode whatever to plot the results.

However, there is a compensating factor. The lower memory modes use one byte to represent several pixels, thus addressing pixels requires extra calculation – whereas the higher memory modes use only one byte to represent a pixel so addressing pixels is much quicker. This means that graphics will be quicker in higher modes.

Mode	F/N loop (secs)	Draw (secs)	Mem. (k)	Colours
0	14.9	51	20	2
1	14.9	38	20	4
2	15.6	34	40	16
3	15.6	–	40	Text
4	14.9	38	20	2
5	14.9	32	20	4
6	14.9	–	20	Text
7	15.6	–	80	T-Text
8	15.6	54	40	4
9	15.6	40	40	16
10	17.1	39	80	256
11	15.6	–	40	Text
12	17.1	61	80	16
13	17.1	45	80	256
14	17.1	–	80	Text
15	21.3	79	160	256
16	19.6	–	132	Text
17	19.5	–	132	Text
18	15.4	70	40	2
19	16.9	77	80	4
20	21.0	98	160	16

• **OS_Pretty Print.** Have you ever wondered how the operating system manages to display all its messages without any of the words getting split at the end of a screen line, no matter what mode it is in? Well, the answer is 'OS_PrettyPrint'.

If you print all your strings using this call it will stop any words going over the end-of-line boundary. This call recognises the following control characters in a special way:

– CR (CHR$(13)) causes not just a carriage return but also a newline.

– TAB (CHR$(9)) causes a tabulation to the next multiple of eight columns.

– CHR$(31) is a 'pad character', that is, the procedure will print a space when this code occurs but it will not break the string up at this point.

Example:

```
  A$="Hello my name is..."
  SYS "OS_PrettyPrint",A$
```

• **The QUICK option** When you use the (Q)uick option in the *COPY and *BACKUP commands, the OS will use all available memory. This means that if you are in high memory screen modes then you will have less memory to use. So if you want even quicker *COPYing or *BACKUPs then try it in a mode which uses less memory.

• **Run$Path and File$Path.** The FileSwitch uses two system variables called Run$Path and File$Path. You can see their values by typing either:

```
  *SHOW File$Path <return> and/or
  *SHOW Run$Path <return>
```

You will probably get the following results:

```
  File$Path : type String, value :
  Run$Path : type String, value : ,%.
```

These are the default settings. The values stored in the File$Path and Run$Path variables are actually a list of directory filenames separated by commas and terminated with dots.

When the FileSwitch is told to read a file it will look at File$Path variable and search for the file in each of the directories listed until it finds a match. Similarly the FileSwitch will do the same when it is told to execute a file, only it will use the Run$Path variable as the list of directories. For example:

```
*SET File$Path RFS:%.,,%.
```

This would cause the FileSwitch to search for the file first in the RFS filing system's library directory, then in the current filing system's current directory (hence the " , , " which means don't add anything to the filename as entered), and finally in the current filing system's library directory.

Here is a list of all the possible directory prefixes (although you can actually specify any directory by name – wildcards may be used):

*	–	all
$	–	root
&	–	user
@	–	current
^	–	parent
%	–	library
}	–	previous

• **Flushing the Mouse.** You may have noticed that the mouse has a buffer of its own. This means that the Arc stores all the mouse movements and clicks just as it stores the keyboard entries. How many of you play around with the mouse when a program is 'thinking' or waiting for a screen to load? This will leave a whole load of mouse positions and clicks in the buffer, so any subsequent MOUSE X,Y,B readings will not reflect the true position of the mouse but the positions stored in the buffer!!! So how do you clear the buffer? To flush all the buffers, type:

```
*FX 15 <return>
```

to flush just the mouse buffer, type:

```
*FX 21,9 <return>
```

• **Local DATA statements.** (Based on an idea sent in by C.R.Fitch) In answer to the query last month, it **is** possible to have DATA statements in a program that does not rely on line numbers. What you can do is to use the error handler to pin-point a line number just before the local data statements. For every data set you want to access, just use the following format:

```
DEFPROCdata_set
LOCAL ERROR
ON ERROR LOCAL RESTORE ERL
IF ERL=0 THEN ERROR 1,"get line
                          number"
RESTORE ERROR
DATA 1,2,3,4,5 : REM put your
                     own data here
ENDPROC
```

Then if you want to use the data set, just call the procedure, as for example:

```
PROCdata_set
FOR i=1 TO 5
READ data:PRINT data
NEXT i
```

The procedure leaves everything exactly as it was (except the data pointer) so it does not affect the program at all. The only snag is that we don't know where the old data pointer was so we can't continue reading data from where we left off! Any ideas??

(The remaining information in this section is about using the MS-DOS emulator. More information than this was sent in, but this was all I could digest with my limited knowledge of MS-DOS. Would anyone be prepared to receive all such information and knock it up into a special MS-DOS section each month?)

MS-DOS Hints and Tips

• **GETfile and PUTfile** on the MS-DOS emulator are difficult to use if you only have a single drive. However, if you configure a RAM disc with MS-DOS you can then copy MS-DOS files onto it and then form the ADFS files on the physical drive and vice versa to take ADFS files in to the MS-DOS file system.

• Programs that work under the PC-emulator. I've had lots of information about MS-DOS software that **does** work, but has anyone found any software that does NOT work under the PC-emulator?

• With version 1.09 of the PC emulator, if you do the following…

```
*LOAD PC.Emulate 10000
?&137C8=0
*SAVE PC.Emulate2 10000
```

you will find that the memory isn't cleared when you press the reset button.

• **One good book** to help new MS-DOS'ers (sorry!) is "Quick Reference Guide to MS-DOS" by Van Wolverton, published by Microsoft Press, obtainable via Watford Electronics at £3.95. (ISBN 1-55615-025-3)

• From various reports, it seems that **CHKDSK does** work properly though it has to be used with care. It can be used to repair the File Allocation Table (FAT) and repair errors in directories. Files are not necessarily stored contiguously on an MS-DOS disc, so if part of a file is lost then CHKDSK can be used to convert lost chains to files (i.e. write to a disc directory). Generally, CHKDSK by itself is safe. CHKDSK *.* /V is usually safe, but CHKDSK *.* /F is dangerous if you don't know what you are doing! A

Monitor Information

• The **Microvitec 1455** (medium resolution – 653 dots horizontally) works OK on the Archimedes giving a quite acceptable picture – "clear, sharp pictures" says one person, "better than the Phillips" says another. Although it has an RGB input for BBC type TTL signals, to use the linear signals from the Archimedes, you have to use the four separate BNC inputs instead: R, G, B and sync. This monitor only works up to mode 17, though.

• The **Electrohome High Res Data Display** monitor from Opus gives "satisfactory results – probably a bit too much green". *(Could it be just the need for a resistor change in the video output of Archimedes? Issue 3, pages 7/8 Ed.)*

• **Philips Monitor/TV Model No 14CF1114** works in that it gives all the colours, but it is not really readable in 132 column mode.

The next two contributions were from Holland, so may not be relevant to UK subscribers.

• The **Philips CM8833** medium resolution monitor works OK on the Archimedes and, with the lead from a cheap walkman type headphone to the relevant pins on the SCART plug, you can get it to produce stereo sound through its speakers! (See below for pin connections.)

• The **Philips CM8873 multi-sync** monitor works well except that in modes 18 to 20, the display shifts horizontally (see the Taxan/NEC review) more than can be compensated for by the external controls. Internal adjustment is needed. The reader says that he does not agree with the PCW review on this monitor which said that the display was unstable. His is rock-steady.

Ben Bles from Woerden, Holland, who sent in the previous comment, also sent in the following:

SCART stands for "Syndicat des Constructeurs d'Appariels de Radio et de Television". (I'll bet

Multi-sync Monitors – NEC versus Taxan

Which Multi-sync?

Just in case you are fortunate enough to be able to afford a multi-sync monitor, I've got hold of two to try – NEC's new Multi-sync II and the Taxan 770 Plus. (The latter was on load for review; grateful thanks to Taxan UK!) I've been able to put them together on the bench and compare them side-by-side. Here are my views and comparisons under various headings.

Price

The RRP's are: Taxan £699 +VAT and NEC £649 +VAT. (£746 and £803 respectively.) However, if you shop around, you should be able to get them somewhat less than that. For example, Beebug members can get the NEC for £635 inc VAT and Watford sell the Taxan at £570 inc VAT.

Specification

Both are 14" monitors for use on CGA, EGA, PGA and MDA standards (i.e. you can use them on your IBM's as well as the Archimedes). Both have analogue and TTL inputs. Both scan roughly 15 – 35 kHz vertically and 50 – 90 Hz horizontally. Both have 0.31 mm dot pitch and both claim a 30 MHz bandwidth. Both weigh about 31 lb (14 kg). So what's the difference? It looks as if they should be exactly the same.

In use

When using the two side by side, certain differences did appear. The basic (subjective) resolution is about the same, but the Taxan didn't have much to spare on either contrast or brightness. In very bright conditions that might be important, though in darker conditions, the NEC's brightness was almost too much even when on minimum.

The main disadvantage of the Taxan is that in the hi-res modes (18, 19, 20) the whole display shifts over to the right by 14 mm (over half an inch). It's not just a fault on the particular monitor I've got because one of the readers reported the same thing – 16 mm in his case. Taxan told him that it must be a fault with the computer, but the NEC monitor seems to cope. I spoke to Taxan's technical bods who, so far, have not come up with a solution.

(Continued from overleaf...)

you wish you'd never asked!) The French equipment manufacturing industry set up this standard for all video equipment, though obviously it is not adopted universally. The SCART connections are:

1	audio out (R)
2	audio in (R)
3	audio out (L)
4	audio ground
5	blue ground
6	audio in (L)
7	blue video (sic!) (sick?)
8	switch voltage (0=TV; 1=accessory)
9	green ground
10	free
11	green video
12	free
13	red ground
14	free
15	red video
16	blanking signal
17	video ground
18	blanking ground
19	video out (CVBS)
20	video in (CVBS)
21	connector ground

The Taxan had more comprehensive controls for picture size and position. Most notably, it had a horizontal size control which the NEC did not. I wanted to make the NEC's picture fill the screen more fully (I like to feel I get better value for money that way!) but was unable to do so. The only control was a horizontal size switch marked "on" and "off" and it made the picture smaller, not bigger.

The Taxan monitor came with a lead (which costs extra) to allow it to be connected to a BBC Micro – it works beautifully. I tried to use the lead with the NEC monitor, but it didn't seem to want to synchronise with the BBC signal. I thought it should have worked because the pin connections seem to be the same for the two monitors. However, NEC have assured me on the phone that their monitor does work on the BBC micro.

Styling and layout

Both are similarly styled in grey plastic on a tilt and swivel stand. The only slight advantage of the NEC was that most of the external controls were on the front panel whereas the Taxan only has contrast and brightness on the front. The Taxan, however, has four sets of preset controls for the four IBM standards accessible with a plastic tool (provided) for adjusting each separately. Though this does not seem to be relevant to the Archimedes itself, it may influence your purchase if you are wanting to be able to use the same monitor on different computers.

A minus for the NEC monitor comes when you compare the leads that are provided. With the Taxan, you get a mains lead that plugs into the mains output socket of the Archimedes whereas the NEC lead just has bare wires at the end and you have to hunt around for a spare mains plug – and then find a spare socket to fit it in! Having paid several hundred pounds for a monitor, I feel that they could at least have put a moulded plug on it for me!

The video leads were also rather different. The Taxan one had long knurled nuts which meant you could secure the plugs easily without having to find a screwdriver and the plugs had moulded leads which made them look less prone to damage than the NEC ones. These had plastic collars which seemed to be not too securely attached to the metal bodies of the plugs. The Taxan lead also had on it what was referred to in the documentation as a "ferrite collar". The NEC lead didn't have one, but I could not find out whether it actually made any difference.

Which one to buy?

There are obviously various pros and cons, but on balance, I think I preferred the NEC monitor, mainly because of the annoying picture shift in the hi-res modes on the Taxan. If you have an IBM computer as well, you may prefer the Taxan because of its greater adjustability. The other factor is what sort of price you have to pay for each monitor and so the NEC wins out again, although Watford Electronics' price for both is the same, £495 +VAT = £570.

Bulk Purchase

If we can get together five or more people, we can get the price of the NEC Multi-sync II's down to £544 inc VAT and if we can get ten or more, we can get down to £530. These prices however are on the basis of cheque with order and the purchaser has to arrange collection from Norwich. If you are interested, give us a ring as soon as possible.

(Actually, the NEC I have been testing, although bought singly, was purchased at a special price. I'd like to keep it but can't afford to at the moment, so if anyone wants one straight away, this one is going for £540 – "first come first served".) A

Help!!!......

• **Help with the magazine.** Would anyone be interested in editing a **special section** in the magazine **for MS-DOS users**? (MS-DOS'ers?!) Now that we have so much material being sent in by readers, I find it difficult enough editing the main part of the magazine, let alone MS-DOS applications about which I know virtually nothing! I would send the MS-DOS editor all the relevant contributions which he/she would have to knock up into some sort of coherent section for inclusion in the magazine each month. If you are interested, let us know.

What about someone else editing a **Languages' Corner**? Again this is something I know nothing about having been weaned on BASIC and 6502 assembler.

I have also received two or three **programs/articles** about trying to get to grips with the **256 colour modes** and I need someone to look at them all and pull them together into one (or a series?) of articles and programs. This is an area where I don't have the necessary technical knowledge or time, to do the editing. *(What do you mean,"Well, what DOES this editor know about?..." ?!?)* *(It's OK, Adrian says he'll do the 256 colour thing.)*

Because of the problems with the RS423, we don't seem to have many contributions about **communications**, but when we do, how about someone editing that separately? Any other special interest sections that you can think of?

• **Stolen!** Acorn Archimedes + colour monitor! Keeps your eyes open and let Paul Fray Ltd (0223 66529) know if you see them. Computer serial Nº 27-AKB15-1000210 and monitor serial Nº 28-AKF11-1000512.

• One reader asks if someone could write a **partial renumber routine** for BASIC. He accepts that the purists don't like numbered programs and that BASIC V has lots of facilities for avoiding the use of line numbers, but he thinks that with the very large programs that Archimedes can handle, such a routine would be invaluable. (But remember that if you use the ARM BASIC editor, there are times when it automatically renumbers the program for you!)

• Could someone write a **checksum utility** for the program listings?

• People are still reporting **problems getting Viewstore to work** on the Archimedes. Has anyone got Viewstore 1.2 working? Has anyone got the utilities working?

• Does anyone know if the **KAGA Vision III monitor** will work on the Archimedes? Dr Peter Howard of Church Crookham would love to know because he, like so many, I suspect, is trying to convince the "powers that be" that an Archimedes would be a good investment. The knowledge that he could use his existing monitor might just persuade his wife to give in!

• Anyone got any idea about the **AOF (Acorn Object Format)** or the library format? The PRM is less than forthcoming.

• **Program suites.** Has anyone tried to use a number of different programs as a suite, passing variables between them? If so, can you give us some clues about the techniques used.

• **Data transfer with Apricot?** Has anyone tried it? In theory you should be able to read Apricot discs on the Archimedes. Or has anyone had any success with serial transfer? (Ian Hamilton, JK578J)

• Lawson Wakefield writes:

"Is there any way to disable the annoying "Finished after nn seconds" message which Arthur keeps coming up with? If you are using Fortran or C, virtually every command is punctuated with the confounded thing!

The linker as supplied with Fortran and C comes up with some amazing error messages. Who would have known that "Uncaught Modula 2 exception – User defined" actually meant "Out of memory"?!

It looks as if there is no form of command line substitution for Arthur command files, even though we now have a powerful command language. In other words, we can't write operating system procedures with variable command line arguments, like %1, %2 etc as used in DOS or the $1, $2 etc under Unix. I know we have aliases but they are only useful for relatively short commands. Perhaps I've overlooked something in the PRM. If so, perhaps someone could enlighten me.

Arthurlib for Fortran? – At present, Acornsoft Fortran offers no interface to Arthur. Is there a version of Arthurlib around like the one supplied with 'C'?"

Help answers

• In the December issue there was a query about machine code programs that would run OK when called from within BASIC but not when *RUN as a separate file. Ken Robbins writes…

"The problem is that *RUN does not provide a stack via R13. R14 IS valid and points to a SWI OS_Exit up in ROM virtual storage. I solved the problem by:

a) Removing the STMFD/LDMFD required for the BASIC linkage and saving R14 in a word of local storage; on exit, load R15 from the 1-word save area.

b) *STAMPing the loadable program and *SETTYPEing as module type &FF8 (absolute code to be loaded at &8000 in the application work-space). Note that offset assembly may be required if a self-relocating programming style is not used."

Let's Name Names!

Well, it had to be in there somewhere! If you do a *MEMORY 3876300 3876585 on OS 1.2, you will see the following information:

Thanks are due to the following people:

Graham Anderson	Nick Reeves
Tudor Brown	Sharon Shelley
Bruce Cockburn	Jim Sutton
Tim Dobson	Stuart Swales
Paul Fellows	Paul Swindell
David Flynn	Jon Thackray
Steve Furber	Alasdair Thomas
Dave Howard	Tony Thompson
Richard King	Hugo Tyson
Richard Manby	Jamie Urquhart
Mike Muller	John Wilkins
Harry Oldham	Jes Wills
Neil Raine	Roger Wilson

Others who have assisted:

Alex Bienek	Mike Hill
John Biggs	Henry Howarth
David Burling	Ian Jack
Martin Clemoes	Kechil Kirkham
Vic Gibling	Andrew Powis
Martyn Gilbert	Jacqui Sanalitro
Mark Hall	Graham Smith

Last and certainly least:

David Bell
Brian Long et Gooney Bird!

That's what it says! Plug in your debugger and have a look if you don't believe me.

Mandelbrot and Julia Sets for Beginners

Barry J. Biddles

You must, by now, have seen some of those amazing Mandelbrot pictures, either in the review in PCW, Aug '87, or on the Acorn demo disk, and perhaps thought that such complicated matters were beyond either your mathematical ability or your computing skill. The purpose of this article is to show that the mathematics is quite straightforward – at the level required to produce the pictures – and that anyone with a pixel plotting command, even in monochrome, can do it for himself.

When I recently introduced this subject at my club, I started by showing a picture, then asking if anyone would like the equation used to produce it. After comments like "My notebook is not big enough!" everyone was surprised to learn that the equation was simply

Z becomes Z^2 + C – applied iteratively.

Ladies first!?

There are two related sets of points, called the Mandelbrot set and the Julia set. I shall start by explaining the Julia set, as it is slightly easier to grasp. Think about the X-Y plane, with a circle of unit radius drawn about the origin. Consider a point Z. Now if the point Z is outside the circle, it must be further than 1 from the origin, and so a point Z^2 will be even further away. On the other hand, if Z is inside the circle, then the point Z^2 will be closer to the origin. If the squaring process is iterated indefinitely, the point will either go to infinity, or will approach the origin.

Before going any further, I must explain exactly what I mean by the square of a point – and this is the slightly difficult part of the maths for a non-specialist. Mathematicians refer to a point Z with coordinates X and Y as (X,Y) or (X Y), but sometimes as X+iY, where 'i' is a marker which signifies that the associated quantity Y is in a direction at right angles to the other one. It turns out to be very convenient if 'i' is more than just a marker; a lot of other maths is simplified if 'i' has the value of the square root of -1. Such a number is imaginary of course, so the quantity Y is sometimes called the imaginary term, and the quantity X is correspondingly called the real term. So if the point Z is

Z: X + iY then the point Z^2 is

Z^2: (X+iY)*(X+iY)

Work it out in the usual way, but remembering that i^2 = -1, and you get

Z^2: (X^2 -Y^2) + i*(2.X.Y)

The two terms in brackets are the new X and Y coordinates of the point Z^2. These are the values after the first application of the process. The next steps all consist of iteration, i.e. feeding the latest 'new values' back into the process for another iteration.

Suppose, now, that you colour the starting point Z according to whether it goes off towards infinity or approaches the origin after many iterations. Use a colour which depends upon the

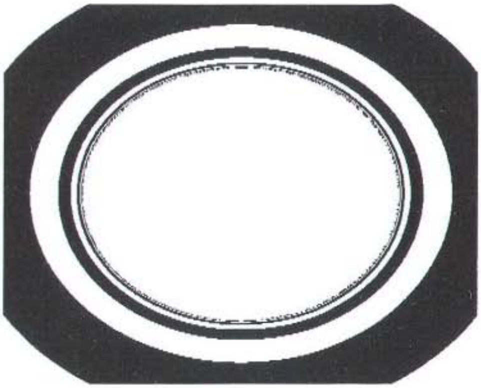


Figure 1

speed with which it takes off toward infinity, but colour it black if it does not. The speed can be assessed by counting how many iterations it required to get well on its way, say to a radius of 2 or beyond (i.e. X^2+Y^2 >=4). What you would get is a black unit circle surrounded by coloured fringes, as shown in Figure 1.

Now we change the rules, to make it more exciting. Instead of the process being simply Z:= Z^2, we use

Z:= Z^2 + C where C= P+iQ

C is what we call a 'complex constant', i.e. a constant made up of a 'real' part and an 'imaginary' part. When you add it to the complex quantity Z you simply add real to real and imaginary to imaginary, so the process now becomes

Z:= (X^2 – Y^2 +P) + i*(2.X.Y + Q)

If P and Q are zero you get the black circle, as before, but if they have non-zero values the effect is quite unexpected. The black disk becomes weirdly distorted, with an infinitely crinkly edge. This kind of shape is called 'fractal' – short for 'fractional dimensions'. It obviously encloses a finite area, yet it has infinite length. It is 'self similar' – meaning that whatever magnification you choose to examine it with you will see much the same level of complexity. Fig 2 shows the appearance when C=0.31+i*0.04. Note that it is the black points which constitute the Julia Set; the coloured regions outside the set simply add to the beauty of the picture. *(Sorry, but can't afford colour repro! Ed.)*

So why can't we have colours inside as well? The problem is that there is no simple test like the outside 'infinity test' to determine which colour to use. An internal point does not necessarily go to the origin, or to any fixed point inside. It may even land on a repetitive cycle of points. If you don't know where it is going, how can you test to see how long it took to get there? I shall return to this problem later.

To summarise Julia, each point on the screen is processed repetitively by the process Z:=Z^2+C where C is a complex constant. The points are coloured according to how fast they go off to infinity, or coloured black if they do not.

"Curiouser and Curiouser", said Alice – The Mandelbrot Set

There is an alternative. Instead of processing every point using a constant value of C, what happens if you always process only one point, usually the origin, but explore the effect of different values of C? The values of the real and imaginary parts of C are going to be taken from the screen coordinates of each point to be considered for colouring, so this time let's have the screen called P, Q instead of X, Y. The coordinates of each screen point then become the complex quantity C=P+iQ (C is no longer constant of course, but to keep it understandable I won't change the name.) The starting point, i.e. the origin, is processed by the same law as before, namely Z:=Z^2+C, and again is

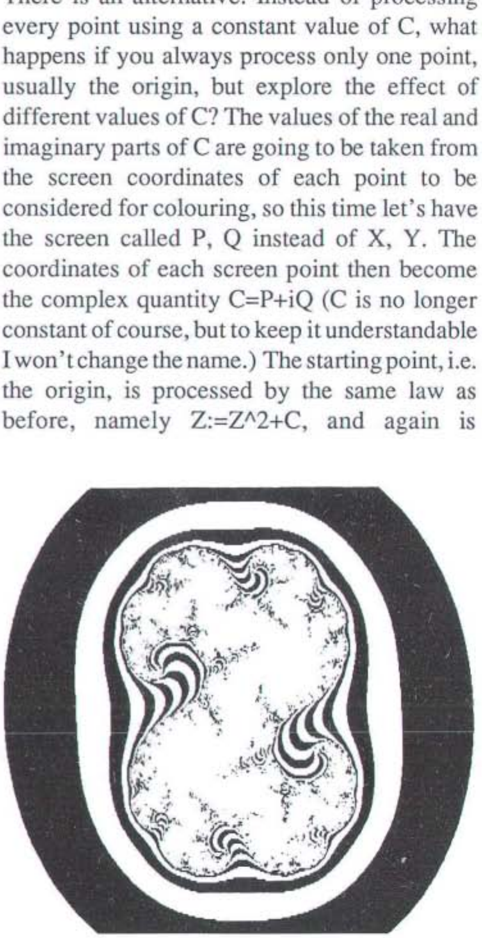


Figure 2

coloured according to how many squarings it takes to get to a distance of 2 or more from the origin, or coloured black if this does not happen.

The well known Mandelbrot 'base picture' is shown in Fig.3, where P spans the range -2.2 to +1.5, and Q goes from -1.5 to +1.5. All of the other Mandelbrot pictures that you may have seen simply use different ranges of P and Q. By specifying very small ranges of P and Q, and displaying them full screen size, you are effectively looking at the edge of the Mandelbrot set with very high magnification. As before, the fractal edge of the set has the property of self-similarity, so that there is a literally infinite amount of detail there to be seen. I typically use magnifications of up to 100,000 with respect to that base picture which I call mag=1 (Compare this with a good optical microscope which would only have a magnification of, say, 1000.)

I find it convenient to specify the pictures according to the coordinates of their centre, and the magnification with respect to the 'base picture'. Fig 4 shows a small detail at -0.74592657, +0.18008656, at a magnification of 8000.

Many of the pictures (when seen in colour) give the overwhelming impression of 'looking back down the beanstalk' to the fields and lakes far below, though this is purely subjective – there is no real association with depth. However, if you zoom the magnification it creates the impression of diving down into those fields, so...

...a pipedream: just imagine a Cray in the living room, creating the pictures at video speed, with the parameters controlled by a joystick for forward, side and 'height', flying through Mandelbrot space! I can't afford a Cray, but I am setting up a similar project based on a few thousand slides and a slide dissolve projection system, entitled 'Mandelbrot – The Movie'.

There are many programs now available to display the Mandelbrot and Julia pictures. Some of them are in machine code, and the best that I have seen takes only 27 seconds per picture, though if it is only integer arithmetic it runs out of resolution at the high magnifications I mentioned earlier. However, these programs are great for exploring the set within those limits.

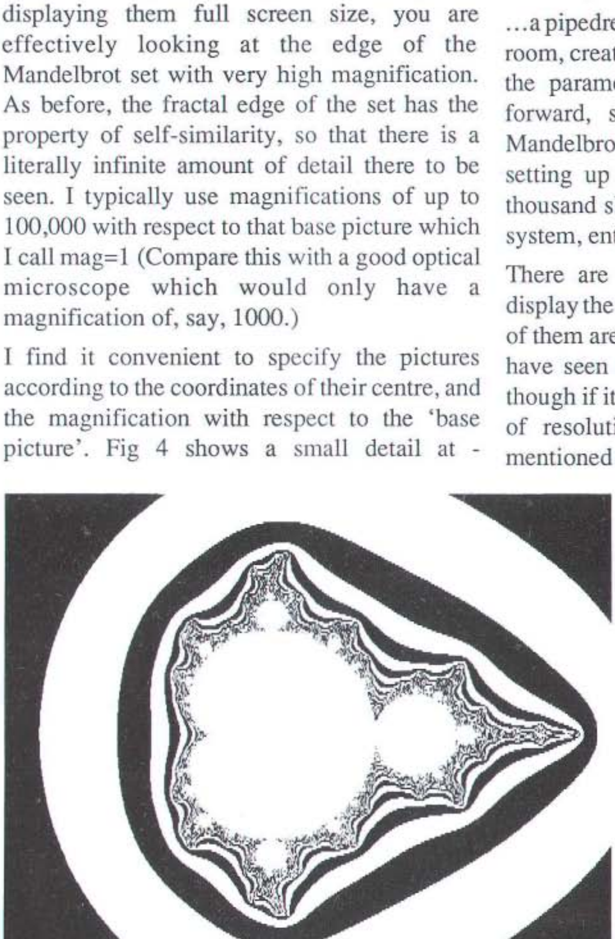


Figure3

My own programs are written in BASIC and in C, and are easier to experiment with, though slower. For example, returning to the problem of colours inside the set, it occurred to me that you do not need to know where the point has gone to, you only need to know when it has arrived, i.e. when it has stopped moving. So if you work out the move length for each iteration of the process, you can test for when it has stopped (effectively) and colour it accordingly. This does indeed produce coloured fringes in the set, but they are not the same ones as shown in the literature, so I guess that this approach is far too simple. I am continuing to work on it.

Where next?

I can't end an article on this subject without referring to the best book that I have yet encountered in this field. Not Mandelbrot's own 'The Fractal Geometry of Nature', which is a very abstruse work suitable only for advanced mathematicians, but 'The Beauty of Fractals' by H.O.Peitgen and P.H.Richter, published by Springer Verlag. Written by a team from Bremmen University, and including specialised articles from a number of people including Mandelbrot himself. However, 80% of the book is still too heavy going for ordinary mortals. (The superficial mathematics is very simple as I have shown, but it leads naturally into topics like the statistics of rough surfaces and very advanced set theory.) However, the book is magnificently illustrated with colour photographs on very good quality art paper, and also contains a brief 'Do-It-Yourself' section, from which I was able to make my start. This is a fascinating book which anyone interested in the subject MUST see, though at over £30 you may have to think twice about buying it.

My BASIC and C programs – including source code – are available free by sending me a formatted disk with return postage and reuseable packing (don't wrap it to withstand nuclear attack – a jiffy bag is quite ample.) Send to Mr B J Biddles, 3 Acer Road, Biggin Hill, Kent. A

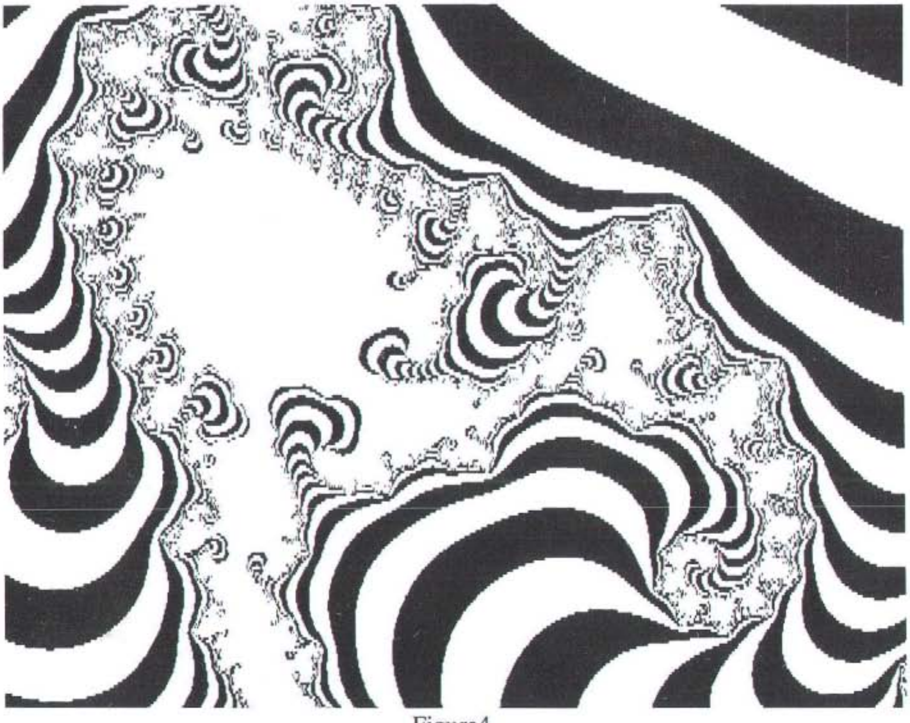


Figure4

ARM Code Mandelbrot Program

William Doggett

William has written a machine code Mandelbrot program, independently of the one in Acorn User. The speed is phenomenal and it tops the Acorn User program by providing more comprehensive facilities – for example, if you start to zoom into an area and change your mind, you can stop the calculation before it has reached the end of the screen and/or zoom back out again to the previous view.

Program Notes

(In these notes the zoom area is defined as the area within, and including, the boundary of the movable square.)

All control is via the mouse, with the controls explained on-screen.

The program starts up with the initial screen, which displays the entire Mandelbrot set. Calculation time is approximately 80 seconds.

Plotting can be interrupted at any time (the current line being completed first), by pressing the <left> button (<select>). The zoom facility is then entered.

The program allows very comprehensive exploration of the Mandelbrot set, by providing a ZOOM OUT as well as a ZOOM IN option.

The ZOOM procedure can be summarised as follows;-

1) Enter zoom facility by pressing the <left> button.
2) Move the mouse to position the zoom area at the region of interest.
3) Scale the zoom area to the required size, by pressing the <left> and/or <middle> buttons.
4) FIX the zoom area at this position by pressing the <right> button.
5) Press <left> or <middle> button to select ZOOM IN or ZOOM OUT respectively. Or press <right> button to QUIT selected region and re-position the zoom area.

The zoom facility can generate two types of error. Both errors are indicated by a bong sound, and the automatic re-positioning of the zoom area.

The first error is produced when an attempt is made to zoom in to an area that is too small to be represented by the program's current position.

The second error is produced when an attempt is made to zoom out to an area, that is too large to be represented by the program.

The user may press <escape> at any time, to return immediately to the initial screen;

The Quit option has been removed from the zoom facility and replaced by CONTINUE. This allows the user to abort the zoom facility and resume plotting the current screen.

Setting up the program

1) Enter *CONFIGURE SPRITESIZE 10 and then press <ctrl-break>.
2) Enter the program and SAVE to disc.
3) In immediate mode, enter PROCsetup <return>. This will plot the initial mandelbrot screen and SAVE it to disc as a sprite.
4) If PROCsetup is not required, it can be deleted and the program re-SAVEd.
5) RUN the program. After a few seconds you will be presented with the initial mandelbrot screen.

```
  10 REM > $.mandelbrot
  20 REM -- By William Doggett --
  30 REM --        Leeds         --
  40 REM --     Version 2.3      --
  50 REM --    (c)  07/02/88     --
  60 MODE 13:PROCinit
  70 ON ERROR PROCerror
  80 PROCmain
  90 END
 100
 110 DEFPROCmain
 120 COLOUR 30:CLS
 130 PRINTSPC72"PRESS"''"LEFT"''
            "MOUSE"''"BUTTON"''"FOR"
                ''"ZOOM"''"FACILITY"
 140 CASE TRUE OF
 150   WHEN b%=0
 160   x=-2:y=-1.25:s=2.5
 170   *SCHOOSE mandelsprite
 180   CLG:PLOT &ED,0,0:Y%=1024
 190   WHEN b%=1 AND Y%<1024
 200   C%-=Y%DIV4*D%
 210   PROCmandelbrot(Y%)
 220   WHEN b%=2,b%=4
 230   !scrnstore=&1FD8000
 240   CLG:PROCmandelbrot(0)
 250 ENDCASE
 260 PROCdelay(50):*FX 21,9
 270 REPEAT MOUSE x%,y%,b%
 280 UNTIL b%=4
 290 ERROR 999,"ZOOM"
 300 ENDPROC
 310
 320 DEFPROCmandelbrot(A%)
 330 PROCdelay(50):*FX 21,9
 340 FOR Y%=A% TO 1020 STEP 4
 350   MOUSE x%,y%,b%
 360   IF b%=4 ERROR 999,"ZOOM"
 370   C%-=D%:CALL MAND
 380 NEXT
 390 ENDPROC
 400
 410 DEFPROCerror
 420 IF ERR=17 b%=0
 430 IF ERR=999 PROCzoom
 440 IF ERR<>17 IF ERR<>999 PRINT
          REPORT$" at line ";ERL:END
 450 ENDPROC
 460
 470 DEFPROCzoom
 480 LOCAL X%,Y%
 490 GCOL 3,63
 500 REPEAT nx=x:ny=y:ns=s
 510   PROCdelay(50):PROCselectarea
 520   PROCdelay(50):PROCselectzoom
 530 UNTIL b%
 540 x=nx:y=ny:s=ns
 550 i=s/256:D%=FNbi(i)
 560 B%=FNbi(x-i):C%=FNbi(y+s+i)
 570 ENDPROC
 580
 590 DEFPROCselectarea
 600 X%=448:Y%=448:S%=128:*FX 21,9
 610 MOUSETO X%,Y%
 620 RECTANGLE X%,Y%,S%
 630 PROCcontrols(1):PROCarea
 640 REPEAT OX%=X%:OY%=Y%:OS%=S%
 650   MOUSE X%,Y%,b%
 660   CASE b% OF
 670     WHEN 2:PROCdelay(1)
 680     IF S%>24 S%-=4:PROCarea
 690     WHEN 4:PROCdelay(1)
 700     IF S%<1000 S%+=4:PROCarea
 710     IF X%+S%>1023 X%-=4
 720     IF Y%+S%>1023 Y%-=4
 730   ENDCASE
 740   IF OX%<>X% OR OY%<>Y% OR
                       OS%<>S% PROCbox
 750 UNTIL b%=1
 760 ENDPROC
 770
 780 DEFPROCselectzoom
 790 PROCcontrols(2):*FX 21,9
 800 REPEAT MOUSE x%,y%,b%:UNTIL b%
 810 CASE b% OF
 820   WHEN 1
 830   WHEN 2:ns=1024*s/S%
 840   nx-=FNout(X%):ny-=FNout(Y%)
 850   IF FNbi(ns)>FNbi(7) b%=0
 860   IF ABS FNbi(nx)>FNbi(7) b%=0
 870   IF ABS FNbi(ny)>FNbi(7) b%=0
 880   WHEN 4:ns=FNin(S%)
 890   nx+=FNin(X%):ny+=FNin(Y%)
 900   IF INT FNbi(ns/256)=0 b%=0
 910   OTHERWISE b%=0
 920 ENDCASE
 930 IF b%=0 VDU 7
 940 RECTANGLE X%,Y%,S%
 950 ENDPROC
 960
 970 DEFFNin(A%)=A%DIV4*s/256
 980 DEFFNout(A%)=A%DIV4*ns/256
 990 DEFFNbi(A)=A*(1<<27)
1000
```

```
1010 DEFPROCbox
1020 WAIT:RECTANGLE OX%,OY%,OS%
1030 RECTANGLE X%,Y%,S%
1040 ENDPROC
1050
1060 DEFPROCarea
1070 MOUSERECTANGLE 0,0,1023-S%,
                               1023-S%
1080 ENDPROC
1090
1100 DEFPROCcontrols(A%)
1110 CLS:COLOUR 30:PRINTTAB(0,4)
           "LEFT:"TAB(0,14)"MIDDLE:
                "TAB(0,24)"RIGHT:"
1120 COLOUR 40
1130 IF A%=1 PRINTTAB(0,6)"Increase"
   TAB(0,16)"Decrease"TAB(0,26)"Fix"
1140 IF A%=2 PRINTTAB(0,6)"Zoom in"
        TAB(0,16)"Zoom out"TAB(0,26)
                          "Continue"
1150 ENDPROC
1160
1170 DEFPROCdelay(D%)
1180 TIME=0
1190 REPEAT UNTIL TIME>D%
1200 ENDPROC
1210
1220 DEFPROCinit
1230 DIM code 1024
1240 PROCassemble:*SNEW
1250 VDU 28,32,31,39,0
1260 VDU 24,0;0;1020;1020;
1270 OFF
1280 b%=0:*SLOAD $.smandel
1290 ENDPROC
1300
1310 DEFPROCassemble
1320 ncol=0:u=1:v=2:i=3
1330 screen=4:count=5
1340 ocol=6:xcoord=7:word=8
1350 x=4:y=5:x2=6:y2=7:sign=8
1360 lhs=9:rhs=10
1370 destL=11:destH=12
1380 tmp=5:tmp2=11
1390 sp=13:link=14:pc=15
1400 FOR I%=0 TO 2 STEP 2
1410   PROCsourcecode(I%)
1420 NEXT
1430 ENDPROC
1440
1450 DEFPROCsourcecode(I%)
1460 P%=code
1470 [OPT I%
1480 .MAND
1490 STMFD   (sp)!,{link}
1500 LDR     screen,scrnstore
1510 MOV     count,#0
1520 MOV     ocol,#0
1530 MOV     word,#0
1540 MOV     xcoord,#0
1550 .loopX
1560 ADD     u,u,i
1570 BL      iterate
1580 TEQ     ncol,ocol,LSR #24
1590 BLNE    draw
1600 ADD     count,count,#1
1610 TEQ     count,#256
1620 BNE     loopX
1630 BL      draw
1640 ADD     screen,screen,#64
1650 STR     screen,scrnstore
1660 LDMFD   (sp)!,{pc}^
1670
1680 .iterate
1690 STMFD   (sp)!,{4,5,6,7,8}
1700 MOV     x,#0
1710 MOV     y,#0
1720 MOV     ncol,#256
1730 .repeat
1740 ;get absolute values
1750 EOR     sign,x,y
1760 TEQ     x,#0
1770 RSBMI   x,x,#0
1780 TEQ     y,#0
1790 RSBMI   y,y,#0
1800 ;perform  x2 = x * x
1810 MOV     lhs,x,LSR #16
1820 BIC     rhs,x,lhs,LSL #16
1830 MUL     destH,lhs,lhs
1840 MUL     lhs,rhs,lhs
1850 MUL     destL,rhs,rhs
1860 ADDS    destL,destL,lhs,LSL #17
1870 ADC     destH,destH,lhs,LSR #15
1880 MOV     x2,destL,LSR #27
1890 ORR     x2,x2,destH,LSL #5
1900 ;perform  y2 = y * y
1910 MOV     lhs,y,LSR #16
1920 BIC     rhs,y,lhs,LSL #16
1930 MUL     destH,lhs,lhs
1940 MUL     lhs,rhs,lhs
1950 MUL     destL,rhs,rhs
1960 ADDS    destL,destL,lhs,LSL #17
1970 ADC     destH,destH,lhs,LSR #15
1980 MOV     y2,destL,LSR #27
1990 ORR     y2,y2,destH,LSL #5
2000 ;perform  y = 2 * x * y + v
2010 MOV     destH,y,LSR #16
2020 BIC     rhs,y,destH,LSL #16
2030 MOV     tmp,x,LSR #16
2040 BIC     lhs,x,tmp,LSL #16
2050 MUL    destL,lhs,rhs
2060 MUL    rhs,tmp,rhs
2070 MLA    lhs,destH,lhs,rhs
2080 MUL    destH,tmp,destH
2090 ADDS   destL,destL,lhs,LSL #16
2100 ADC    destH,destH,lhs,LSR #16
2110 MOV    y,destL,LSR #27
2120 ORR    y,y,destH,LSL #5
2130 MOVS   sign,sign
2140 RSBMI  y,y,#0
2150 ADD    y,v,y,LSL #1
2160 ;perform  x = x2 - y2 + u
2170 SUB    x,x2,y2
2180 ADD    x,x,u
2190 ;loop again?
2200 ADD    lhs,x2,y2
2210 CMP    lhs,#FNbi(4)
2220 BHS    return
2230 SUBS   ncol,ncol,#1
2240 BNE    repeat
2250 .return
2260 LDMFD  (sp)!,{4,5,6,7,8}
2270 MOV    pc,link
2280
2290 .draw
2300 MVN    rhs,#0
2310 ANDS   lhs,xcoord,#3
2320 BEQ    over
2330 MOV    tmp2,lhs,LSL #3
2340 BIC    word,word,rhs,LSL tmp2
2350 ORR    word,word,ocol,LSL tmp2
2360 RSB    lhs,lhs,#4
2370 ADD    xcoord,xcoord,lhs
2380 CMP    xcoord,count
2390 STRLS  word,[screen],#4
2400 BHS    finish
2410 .over
2420 MOV    word,ocol
2430 .loop
2440 SUB    lhs,count,xcoord
2450 CMP    lhs,#8
2460 STRHS  word,[screen],#4
2470 STRHS  word,[screen],#4
2480 ADDHS  xcoord,xcoord,#8
2490 BHI    loop
2500 SUB    lhs,count,xcoord
2510 CMP    lhs,#4
2520 STRHS  word,[screen],#4
2530 ADDHS  xcoord,xcoord,#4
2540 .finish
2550 MOV    xcoord,count
2560 ORR    ocol,ncol,ncol,LSL #8
2570 ORR    ocol,ocol,ocol,LSL #16
2580 MOV    pc,link
2590
2600 .scrnstore
2610 EQUD 0
2620 ]
2630 ENDPROC
2640
2650 DEFPROCsetup
2660 LOCAL ERROR
2670 ON ERROR LOCAL PROCsprite:END
2680 MODE 13:PROCinit
2690 ENDPROC
2700
2710 DEFPROCsprite
2720 x=-2:y=-1.25:s=2.5
2730 i=s/256:D%=FNbi(i)
2740 B%=FNbi(x-i):C%=FNbi(y+s+i)
2750 !scrnstore=&1FD8000
2760 CLG:PROCmandelbrot(0)
2770 MOVE 0,0:MOVE 1023,1023
2780 *SGET mandelsprite
2790 *SSAVE $.smandel
2800 ENDPROC
```

MAXGAMMON

Backgammon for the Archimedes

Makes full use of the WIMP Environment
Runs as an Archimedes desktop accessory
Computer plays either or both sides
Options to save and load games
Sophisticated wind/rewind controls
Written in BASIC and ARM Assembler

Only £9.50 inclusive

Send cheques to :

MAXIMUM
44 Manor Road,
Wokingham,
Berkshire,
RG11 4AR

New Life!?

Modifications to last month's Life program

There was a mistake in the "tumbler" picture which was supposed to be an oscillator. It should have had an extra block at each of the two bottom corners.

If you want to try some of the larger patterns, Tony Brain has given us some modifications which make the program work with a field of 160 by 120 cells. If you typed in the program from the magazine then the following lines need to be typed in. Note that the blank lines are significant – they delete existing lines that are no longer needed.

```
  10 REM >Life_160
  15 REM Alterations to Life to give
                      160 x 120 field.
 320 DIM code% 25000,osblock% 16
 570 FOR no%=0 TO 19516 STEP 4
 660 ?(array%+1770+RND(140)+
                     (160*RND(100)))=1
 810 X%-=X% MOD 8:Y%-=Y% MOD 8
 990 GCOL4,0:MOVEX%-4,Y%-4
1000 PLOT1,12,0:PLOT1,0,8
1010 PLOT1,-12,0:PLOT1,0,-8
1150 GCOL2:RECTANGLE FILL pos%,0,
                              len%,63
1360 POINT X%,Y%:POINT X%+2,Y%
1480 =array%+(X%/8)+(128-Y%/8)*160
1510 rows%=120:columns%=160
1580 live_no=11:sp=13
1630 EQUD 0:EQUD 1:EQUD 2:EQUD 160
1640 EQUD162:EQUD320:EQUD321:EQUD322
1645 .end EQUD end%
1680 ADR pointer,end
1690 LDR array_end,[pointer]
1695 SUB array_end,array_end,#320
1730 MOV loop_count,#28
1750 LDR offset_value,[pointer,
                           loop_count]
1790 SUBS loop_count,loop_count,#4
1800
1810 BGE count
1820 LDRB cell_value,[array_addr
                                ,#161]
1920 STRB cell_value,[array_addr
                                ,#161]
2050
2080 .display
2120
2140 ADD array_addr,array_addr,#160
2170
2260 MOVEQ colour,#&11
2270
2280
2290 STRB colour,[scrn_addr,#320]
2300
2310
2320 ADD scrn_addr,scrn_addr,#2
2350 ADD scrn_addr,scrn_addr,#320
2380 STR live_no,number
2430 P%+=19520
```

If you bought the program disc, you will have a version of the program which we modified to allow you to load and save patterns on disc. If you have that version then these are the modifications needed.

```
  10 REM >Life_160
  15 REM Alterations to Life to give
                      160 x 120 field.
 340 DIM code% 25000,osblock% 16
 590 FOR no%=0 TO 19516 STEP 4
 680 ?(array%+1770+RND(140)+
                     (160*RND(100)))=1
 830 X%-=X% MOD 8:Y%-=Y% MOD 8
1010 GCOL4,0:MOVEX%-4,Y%-4
1020 PLOT1,12,0:PLOT1,0,8
1030 PLOT1,-12,0:PLOT1,0,-8
1170 GCOL2:RECTANGLE FILL pos%,0,
                              len%,63
1380 POINT X%,Y%:POINT X%+2,Y%
1500 =array%+(X%/8)+(128-Y%/8)*160
1760 OSCLI("SAVE $.LIFE.SCREENS."+
       filename$+" "+STR$~(array%)+"
                         "+STR$~(end%))
2360 rows%=120:columns%=160
2430 live_no=11:sp=13
2480 EQUD 0:EQUD 1:EQUD 2:EQUD 160
2490 EQUD162:EQUD320:EQUD321:EQUD322
2495 .end EQUD end%
2530 ADR pointer,end
2540 LDR array_end,[pointer]
2545 SUB array_end,array_end,#320
2580 MOV loop_count,#28
2600 LDR offset_value,[pointer,
                           loop_count]
```

```
2640 SUBS loop_count,loop_count,#4
2650
2660 BGE count
2670 LDRB cell_value,[array_addr
                              ,#161]
2770 STRB cell_value,[array_addr
                              ,#161]
2900
2930 .display
2970
2990 ADD array_addr,array_addr,#160
3020
3110 MOVEQ colour,#&11
3120
3130
3140 STRB colour,[scrn_addr,#320]
3150
3160
3170 ADD scrn_addr,scrn_addr,#2
3200 ADD scrn_addr,scrn_addr,#320
3230 STR live_no,number
3280 P%+=19520
```

This month's program disc contains the modified programs plus a number of classical patterns for you to try out for yourself.

Bug or Feature?

• 'OS_SpriteOp': As you may well have noticed, the sprite editor does not work properly on 1.2 OS and the reason for this is a bug is 'OS_SpriteOp' call. This bug seems which has been introduced in 1.2. It returns bogus values for some colours when used to get the colour and tint of a pixel.

• In modes 18 to 20, the pointer does indeed 'break up' when it reaches the right hand side of the screen – I don't know if this is a hardware fault or software. (I'll tell you when I get my 2.0 OS!) Could it perhaps be linked with the 'shift to the right' which occurs on some multi-sync monitors when you go into the hi-res modes?

• Problems with TRACE – this was reported last month, but one reader tells us that Acorn say this is 'not a bug, but a result of the speed of BASIC 5.' Hmm!?

'C' Book Review

"The C Book: featuring the draft ANSI C standard" by Mike Banahan, published by Addison Wesley at £15.95. (ISBN 0-201-17370-0)

Anyone who has purchased Acorn's C compiler for the Archimedes will find this newly published book extremely useful. Acorn's implementation of C conforms very closely to the latest ANSI draft standard for the language but the user guide gives very little description of the language and most available books on C were written well before the draft standard and are seriously out-of-date.

Mike Banahan's book gives a lot of its library functions in such a way that it can easily be used as a 'user guide' for C on the Archimedes (or other new standard-conforming C compilers, such as Microsoft C version 5 for IBM compatibles). The style of the book is such that not only is it very readable for those who already know earlier versions of C but it would also serve as an excellent introduction for anyone who can program in BASIC, Pascal, Fortran or assembler but is new to C.

Colin Deans.

Readers' Comments

A chance here for readers to put their own view about anything vaguely to do with Archimedes and/or Archive. Preferably only polite and constructive comments, please!

• DFS to ADFS utility. One reader comments that having attached a 5.25" disc drive to the Archimedes, CJE Micro's utility for reading DFS discs puts the serial link set-up into the stone age!

FORTRAN 77 on the Archimedes

Leslie and Gwyneth Pettit

As Fortran users for 28 years we were eagerly anticipating our Archimedes 440 with the Acornsoft Fortran 77 compiler. Have we been disappointed? The short answer is No; the machine and the implementation of Fortran 77 are both excellent. The documentation and graphics extensions fall a long way short, however, and the speed of compiled programs is disappointing as a result of the absence of a maths co-processor. The actual compiler is very fast and efficient compared with other micro implementations of Fortran and the final object code is comparatively compact. The major problem comes in actually implementing the necessary commands, especially when using a hard disk system, and in compensating for the shortcomings of the documentation.

Perhaps only experienced Fortran users will want to use Fortran 77 on the Archimedes, but even they will need more information on libraries and the linking stage, particularly if working from a hard disk. If you already have a 300-series Archimedes and are considering buying the Fortran 77 compiler, we are not claiming that the Archimedes 440 is essential for Fortran, but its 4 Mb memory and the hard disk are an ideal Fortran environment.

Terms of Review

We have compared Acornsoft Fortran 77 with other micro-computer implementations – specifically, with Acorn 32000 Fortran 77 under Panos on the Acorn 32016 co-processor and with IBM Professional Fortran and Prospero F77 on IBM compatible micros. These are the Amstrad 1640 (8086 CPU) and the Elonex (NEC V27 CPU), both having the 8087 co-processor and a 20 M hard disk. We also have experience of using Fortran on mainframes (CDC 1604 and Amdahl) and mini-computers (VAX and Data General NOVA) though these are relevant only as background experience.

This review divides into two parts – some helpful advice, to make the first steps for new users less traumatic, followed by quantitative comparisons (the term benchmark can be misleading) on Archimedes Fortran 77 compared with other micro versions.

First Steps with Fortran on the Archimedes

Acornsoft Fortran 77 arrives on a 3.5" floppy disk with a slim spiral bound handbook and instructions for separating the compiler and linker onto two floppy disks for normal use. There are no explicit instructions for putting it on a hard disk (as are provided with the Acorn 32016 languages) which would be helpful for users who are not even familiar yet with ADFS.

There is no editor included, although it is essential to be able to enter Fortran source code in text format, or edit it after transfer from another machine. Advertisements claim that ArcWriter is provided with the Archimedes 440 – it has not yet materialised ! However any text editor (e.g. Wordwise, View or Edit) will do, provided you are prepared to replace line feed characters (&0A) with carriage returns (&0D) if you are working from within the 6502 emulator. By the way, you may like to know that you can exit the emulator to return to your working directory with the command *QUIT.

Fortran for Beginners

Before we tackle specific problems of Fortran on the Archimedes, we shall outline the way it operates, for those who may be unfamiliar with a compiled language. We shall assume you have become familiar with ADFS and its directory structure, and we shall first assume that you are using Fortran from the floppy disk drive.

The Fortran 77 compiler and linker use directories f77 (for the source code), AOF (for the object-code Archimedes Object Format files) and TMP (for storing temporary code). They also require a directory LIBRARY for the necessary libraries etc. and a directory EXECLIB to hold command files.

Compilation of Fortran source-code to an object-code version in directory AOF takes 2 passes, using the 'Front End', called f77fe, for the first pass followed by the 'Code Generator', called f77cg. Intermediate code is stored as a temporary file (TMP.fcode) in directory TMP. These steps can be performed sequentially, using the command file f77 provided on the disk in directory EXECLIB. If you only want to check your syntax, however, you can call the front-end pass separately and the manual does describe in adequate detail how to invoke both passes independently.

After compilation, the object-code (AOF) file is linked with the Fortran 77 library using the universal Archimedes linker. Documentation on the linker is very meagre in the Fortran handbook, which points you to the Programmers Reference Manual (Appendix B), but the 'linkf77' command file in directory EXECLIB will supervise the operation provided you do not wish to link more than one AOF file. Most working programs will require the linking of a number of compiled program modules and the Reference Manual suggests this command to link multiple object-code modules named AOF.Prog1, AOF.Prog2 etc. :

```
Link AOF.Prog1, AOF.Prog2,..,
           -library lib.f77 -image
                        <progname>
```

You can also obtain information on link options by typing the command LINK -HELP at the keyboard.

Running Your Fortran Program

The linker generates an executable program 'progname' in the current directory (which is the root directory on the floppy disk as supplied), which can be run by typing the command

```
*progname
```

Working Directories on the Hard Disk

On a hard disk, the root directory can easily become cluttered with out of date files, which make disk housekeeping very tedious. We decided the most efficient structure was to add only the Acornsoft directories EXECLIB and TMP to the root directory, which already has a sub-directory LIBRARY. Directory EXECLIB contains the command files f77 and linkf77, and LIBRARY must have the Fortran library files added. Initially, directory TMP will be empty.

We created a new directory called Fortran and used this as our working directory, with sub-directories f77 (for source-code) and AOF (for object-code). We had to create another sub-directory here called TMP (both TMP directories appear to be necessary for the command files to work). You can, of course, include any other sub-directories you find necessary for efficient storage and retrieval of your source-code files or your fully compiled/ linked programs. The problem of the hard disk was more acute since we were adding the C compiler at the same time, which could have inflated the entries in the root directory to unacceptable proportions.

Having created these directories, and using Fortran as the current directory, the Acorn command files f77 <fname> and linkf77 <fname>, or the command string given above, will work correctly and all file references will find their targets. The final executable program appears in directory Fortran, and is called by the command *<fname> – but never forget that the floating point emulator has first to be loaded with the command *fpe. Once the file has been correctly compiled and linked, you can optionally transfer it to one of your other subdirectories to keep the working environment uncluttered.

Writing your own Command Files for Compiling/Linking

Command strings like the link command shown above can be typed in dynamically or for repeated use they can be constructed via *BUILD <name> (or by using a text editor) and executed with the command *EXEC <name>. If you subsequently set their filetype to &FFE you can execute them by typing just *<name>.

The shortcoming of constructing your own command strings is that each command file must name its target files explicitly, whereas Acornsoft's f77 and linkf77 will accept filenames as parameters. The format of the Acornsoft command files is similar to those of Panos on the 32016, but we cannot find any explanation as to how they are implemented on the Archimedes! Will anyone volunteer to write an article on the use of files in the EXECLIB directory ?

In particular, the Fortran 77 Release Note states that you if you have the command file f77 in directory EXECLIB , you must have a companion file named f77 in the library directory. The format of the companion file, and the connection with the link file, is nowhere explained and even with a 20 Mbyte hard disk, the space for all the companion files, if you have a number of command files, seems profligate.

Acornsoft Command Files

The contents of the command file f77 in execlib are:

```
. Command sequence to compile Fortran
.
.    f77 source [-object obj] [-opt
                         options] [-m n]
.
. Requires directories f77, aof and
                                    tmp.
.
.key from/a,object/k,opt/k,m/k
f77fe f77.<from> -to tmp.fcode -opt
                                <opt$+>
f77cg tmp.fcode -to aof.<object$
          <from> -opt m<m$20><opt$+>
remove tmp.fcode
```

As you see, it uses commands prefixed with a full stop, which are not documented in either the Archimedes User Guide or Programmers Reference Manuals. This format resembles the Panos command language of the 32016 with which we are familiar, but unfortunately it is not identical, so the Panos guide is not sufficient. In particular, the keystring beginning with .key is not defined and the Panos range of keywords and arguments, which is extensive, may not all apply to the Archimedes.

We would have liked to set up, for example, similar keystring command files for each pass (f77fe and f77cg) to be called separately. Operation of fe and cg, which are fortunately clearly documented, can still be simplified in use by setting up aliases – it would have been helpful if the documentation had given some sample alias$ suggestions. Our suggestions are these:-

```
*set alias$fe "f77fe f77.%0 -to
                          TMP.fcode"
*set alias$cg "f77cg TMP.fcode -to
                              AOF.%0"
```

You can then type these brief commands :-

```
fe <progname>
cg <progname>
```

to implement both passes of the compiler; the parameter <progname> is substituted for %0 in the aliases above.

Using Graphics with Fortran 77

Archimedes graphics can be used in Fortran programs, provided they have equivalent VDU commands – but why don't Acorn provide a starter library of graphics subroutines, as they did for the 32016 ? There are no demonstration programs on the disk (apart from the minimal test program 'Hello World') unlike most suppliers of Fortran for the IBM micros who provide a variety of demonstrations. Even the Acorn 'C' compiler has some demonstration software. It is even more unfortunate that the example on using VDU control codes in the Fortran manual contains prehistoric Hollerith characters!

Here then is a sample subroutine to perform a PLOT command – all other VDU commands can be implemented similarly:

```
INTEGER num,x,y
CALL PLOT(num,x,y)
SUBROUTINE PLOT(num,x1,y1)
INTEGER num,x1,y1
WRITE(6,'(6A)') CHAR(25),CHAR(num),
      CHAR(MOD(x1,256)), + CHAR(x1/
             256),CHAR(MOD(y1,256))
                            ,CHAR(y1/256)
END
```

(This routine assumes channel 5 is not configured as a printer channel).

Users not familiar with BBC BASIC and the VDU control codes may not be aware that VDU 25 is equivalent to the BASIC keyword PLOT; num is used here as the PLOT code, where MOVE requires num=4 and DRAW num=5. The same users may also be unaware that since VDU codes are limited to the range 0 to 255, screen codes (which may exceed this range) are sent as <low byte> followed by <high byte>.

Compatibility

Every source program using legal Fortran 77 (or 66) code which we tried out was compiled and linked satisfactorily. The compiler is in fact slightly more rigorous than the IBM compilers, which had allowed unprofessional jumps into IF-structures. Problems arise when your program tries to access the Archimedes operating system to take advantage of the wonderful facilities available there. There appears to be no way of accessing the operating system variables, not even time and date, and the 32016 system calls such as IFCOMMAND LINE are not implemented.

Clearly, taking advantage of sophisticated Archimedes facilities such as sound, window or OSBYTE calls (e.g. to flush the keyboard buffer) is very important if Fortran programs are to exhibit the Archimedes to advantage. If Acorn cannot release details of general access, they could perhaps provide a disk of utility subroutines. It is also essential to be able to incorporate machine-code routines for plotter-drivers and graphics tablets if full graphics performance is to be feasible.

We were further disappointed in a spirited bid to access the operating system indirectly through 'C' AOF files, having been led to believe that the linker could include any files in AOF format, but this only produced a rich variety of obscure error messages. After much experimentation we found, buried in the Programmer's Reference Manual at the bottom of page 636, the statement that C can call Fortran but that Fortran cannot call C. Our only hope therefore lies in a utilities package from some informed source, if we wish to do more than add standard VDU graphics to Fortran programs.

Working Comparisons

Once a program has been written, how does it compare with IBM implementations of Fortran? Firstly the compilation stage is much faster. A typical program unit (360 lines) took the following compilation times:

Amstrad (IBM Professional Fortran)	184 secs
Amstrad (Prospero F77)	83 secs
Acorn 32016 co-processor	98 secs
Archimedes	21 secs

Execution times did not compare so favourably. The following are figures obtained using a powerful program, SUPERQUAD, used in chemistry for calculating stability constants of metal complexes. SUPERQUAD is essentially a least squares minimization program – if you want more details, look up Journal of the Chemical Society, Dalton Transactions, 1985,1196. These were the times required for the solution of 252 equations for 171 unknowns:

Amstrad 1640 (8087 co-processor)	42 secs.
Acorn 32016 co-processor	38 secs.
Archimedes	47 secs.

Code generation does appear to be more efficient on the Archimedes, where the code size of 201,144 bytes was 20% less than on the Amstrad 1640 (253,302 bytes).

Comparison with BBC BASIC

The program SUPERQUAD has been translated into BBC BASIC and on the MASTER 128 (using code overlays) the time required for the same set of equations was 1474 secs. (725 secs. with the turbo card). On the Archimedes in BASIC V, where the code requires no overlays, the times taken were 105 secs. in ROM BASIC and 79 secs. in RAM BASIC. Since these are timings for an executed language, requiring no overheads of compiling and linking, fully structured and permitting free access to the operating system, you could be forgiven for retaining BASIC as your working language, with its advantages of interpretive development, unless you already have Fortran source code written and tested on another system.

In Fortran, Archimedes is Slower...

The quoted timings highlight the greatest weakness of the Archimedes when used for mathematical calculations, since the software floating point emulator is no match for a maths co-processor. Until one is available there is little to be gained speed-wise compared to IBM compatibles, particularly when comparing with the 286/386 machines, unless much time is spent on compiling and linking during development.

...but Bigger

Where the Archimedes does gain (apart from fast compilation and linking) is its potential to hold much larger programs. IBM machines are limited to 640k until the OS/2 system is introduced, while on the Archimedes 440 we have up to 4 Mbyte. It should be possible to transfer from mainframe computers many existing professional and scientific programs (such as those used in chemistry for displaying crystal structures) which, at the moment, cannot be squeezed into IBM micros. With the promise of a full RISC set in the future, including a floating-point processor, an exciting world of applications is open to the Archimedes and its users. A

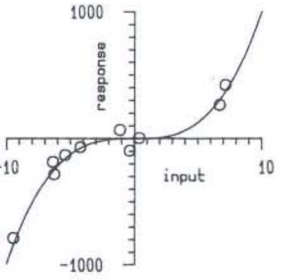
In the light of comments made in the above article, Leslie and Gwyneth, and no doubt many others will be pleased to see the contents of the advertisement opposite. We have been promised a review copy and hope to have some comments in time for the next issue.

New software for the Archimedes

FORTRAN Graphics Library £ 49.50

A library containing over 90 subroutines to add powerful graphics capabilities to the Acorn FORTRAN 77 compiler.

Provides equivalents for BASIC graphics statements (CIRCLE etc). Also includes a comprehensive set of routines for generating sophisticated graphs, histograms, pie charts, etc, either on the monitor or on HPGL-compatible plotters (e.g. HP 7475A).



The XED Text-File Editor £ 19.50

The ideal editor for Pascal, FORTRAN and C programs, command files, etc. Powerful full screen editor using 40, 80 or 132 column screen modes and all ASCII characters 0 - 255, plus comprehensive mainframe-style editing language including sophisticated file input/output and facility to execute command files. Full access to *-commands plus ability to compile and run programs, enter BASIC, etc, while editing, without loss of current text.

Printer Module £ 12.50

Tired of having to wait while files are printed? Load our Printer Module from disc and your Archimedes instantly has a very versatile printer buffer (size 1 Kbyte up to maximum available RAM) that allows you to carry on programming while your printer is churning out paper. Module also has very fast text and graphics screen dumps for Epson-FX compatible printers, and commands to send text files to the printer.

Each package is supplied on disc and includes a comprehensive user guide. FORTRAN Graphics Library and XED Editor both require machine with at least 1 Mbyte of RAM.

Please write for further details, or send your order with a cheque or P.O. payable to 'CCD Computer Services'.
No credit cards please. All prices fully inclusive.
Official orders from Universities, Govt. Labs, etc welcome.

CCD Computer Services

71 Marlborough Park Avenue, SIDCUP, Kent DA15 9DL.
Tel. 01-302 5427.

The Floating Point Emulator

A.J.Smith

There has been talk recently of my current pet project – the Floating Point Emulator. I myself have had a few problems, but these have been mainly due to inadequacies in the Programmers' Reference Manual.

I have (I think) successfully converted Barry Biddle's Julia and Mandelbrot programs into machine code, using floating point arithmetic with the aid of the emulator. Dismayed at the BASIC assembler for not assembling these opcodes (especially since *MEMORYI appears to disassemble them!) I decided to write my own assembler. Unfortunately, after almost completing my masterpiece I realised that the binary format of the CMF/CNF/CMFE/CNFE instructions is not listed. In desperation I turned to Peter Cockerell's otherwise excellent book to find the same information missing. As I am sure you realise, floating point algorithms are rather awkward to write without being able to do comparisons of floating point numbers.

My problems were compounded when the Debugger (Arthur 0.3) refused to disassemble correctly. The version supplied with Arthur 1.2 is better, but still incorrect! Even still further, page 580 of the Reference Manual has a table headed "e" and "f". Replacing these headings with "g" and "h" respectively certainly aids assembling, whether by assembler or by hand.

I have since done some painstaking trial-and-error work to determine the binary format of the compare instructions – with a bit of help from the output of the ANSI C compiler. The format as I have worked it out is:

Bits	Contents
31-28	<cond>
27-24	1110
23-21	<abc> (opcode) – see below
20-19	10
18-16	<Fn>
15- 4	1111 0001 0001
3	<i> (immediate) – see below
2-0	<Fm> or <immediate value>

Notes

<cond> and <immediate value> have the same values as for all other F.P. instructions. This is where the manual can be believed!

<abc> can take the following values for operand:

Opcode	abc
CMF	100
CNF	101
CMFE	110
CNFE	111

<i> should contain 0 to indicate that <Fm> is a floating point register, and 1 to indicate that <Fm> denotes a floating point constant.

More untruths from the manual: Page 583 gives the syntax for the compares. Replace "op <cond>prec<round> Fm, Fn" with the rather different "op<cond> Fn, (Fm #value)".

Before I leave the subject altogether, in trying to decipher various manuals, I managed to get the debugger to disassemble "CDP" and "MCR" instructions (which use registers C 1 etc!). Further research (ARM Assembly language programming page 180+) shows that these are further co-processor instructions (i.e. non floating-point). I wonder what other co-processor emulators Acorn have up their sleeves?

Using the Debugger Module

Gerald Fitton

Introduction

The Acorn Debugger is a Relocatable Module provided on disc for OS 0.2 and in ROM for OS 1.2. Although the utilities it contains are most useful to the assembly language programmer, it is wrong to assume that the debugger module is of no value to the BASIC programmer. In these articles we shall have a look at a BASIC application or two which might be useful.

For the assembly language programmer, one serious omission is the inability to modify machine code by entering an assembler mnemonic: you can enter only the machine code instruction as a hexadecimal number. Perhaps we can provide a fix for that. Those of you who have followed my previous articles will know that I believe in writing self-contained functions and procedures. This sort of FN or PROC can be used as a building block for the construction of quite complex programs which can be built up in easy stages. Hopefully, you'll find something of use to you in this series of articles, even if you don't want to do any debugging of assembler or machine code programs.

The *Memory Command

This command is available in three forms, to display memory, to disassemble a machine code program into its assembler mnemonics and a way of entering hex code into memory. This month we shall deal with the first form only and illustrate its use from within BASIC with a program called Memory01.

Program Description

This program uses the Debugger *Memory command to display a block of memory. However, from within the BASIC environment, it also allows the user to modify any byte of memory by entering the new value either as a printable character from the keyboard or a hexadecimal code. With the parameter type$=" B", the procedure PROCmemory, at line 500, will display at screen (text) co-ordinates (x%,y%) a block of memory of length=length% in bytes starting at start%.

At line 110, the global variable start% is set to the start of the buffer declared in the previous line. There is a REM on line 210. You could insert a line here which loads a disc file into the buffer, use the program to modify it and then save the modified version (line 270).

To see what is in the disc file called "buffer", a *DUMP is used at line 320. PROCmemory is called from within FNmodify at line 1370.

The rest of the program, FNalterASCII and FNchangehex, together with their respective PROCinserts, allow the user to change the contents of the buffer (or any other part of memory as we shall see) in either ASCII (printable keyboard characters) or hex code (which can be unprintable – no comments from Ed, I hope!). There is a lot of duplication between FNalterASCII and FNchangehex and they could, no doubt, be combined into one function but the result would not have been as clear.

What Can You Do With The Program?

Having typed in the program, or loaded it off the Archive program disc, make sure you have a version on disc which has LR attributes. (*ACCESS MEMORY01 LR<return>) When you start playing with memory it is all too easy to crash the system or write not so funny things to disc!

With the version on the Archive program disc, when you run the program you will display a 256 byte block of memory from &9FD4, which is the start of the buffer. If you have typed it in yourself, then you might have an odd space here or there which will mean that buffer% is somewhere slightly different. In any case, what you see on the screen will be the 256 byte buffer located in memory at buffer%. Type in a few characters and then press <return> to save the buffer to disc. Having got a buffer file you can now modify line 210 to read OSCLI("*LOAD buffer "+STR$~(buffer%)):REM Loads buffer file to buffer%. When you run the program you may find that the buffer address has changed a little (but not if you type in 210 exactly as written above including the REM).

Press <tab> to toggle between ASCII entry and hex entry. You will notice that the position of the cursor changes to the hex part of the display. If you type in 41 then a letter A will appear in the ASCII section of the screen display. This is because the ASCII code for A is &41. You can move through the memory with the cursor (arrow) keys, or more rapidly by holding down the <shift> key at the same time as you tap the cursor keys (lines 680 and 1030).

My printer is an Epson FX80+ and I like producing my listings in Elite (12 characters per inch) with a 10 character margin. Because my ribbon is a bit worn, I use double strike. The codes to be sent to the printer to produce this effect are 27, 71, 27, 77, 27, 108, 10 in decimal, or in hex 1B, 47, 1B, 4D, 1B, 6C, 0A. RUN the program, press <shift> with the up cursor followed by <tab> and you should then be at the start of the buffer (&9FD4) and in hex mode. Type in the hex codes "1B, 47, 1B, 4D, 1B, 6C, 0A" then <tab> into the ASCII section and type a few words. Make sure the rest of the buffer is hex zeros, and then save the file by pressing <return>. The program:-

```
    10*FX 3,10
    20*PRINT buffer
    30*FX 3,0
    40END
```

will send the buffer file to the printer complete with the "Escape" codes to produce double strike, Elite printing with a left margin of 10 characters. It is probably a bit of an exaggeration to call such a buffer file a printer driver, and this program will never replace a good word processor, however I'm sure you can see the potential of the method. For example, instead of just a file editor, the buffer could be loaded and saved with the command SYS "ADFS_DiscOp" and become a disc editor. Make sure you make the buffer large enough at line 90.

Use `[SYS "ADFS_DiscOp",,reason%, discaddress%,buffer%,buffersize% TO check%]`

The parameter reason%=1 to load from disc or 2 to write to the disc; the parameter discaddress% is the address in bytes from the start of the disc.

A Program Modifier

I think I am a "Morning Person". That's why I work on recursive procedures (FNs or PROCs that call themselves) in the morning: it needs a clear head! Even harder are programs that modify their own code. Now I believe that self-modifying code is generally bad programming and, now that we have got beyond the stage of computers with more than 1kbyte of memory, there is little excuse for such 'party tricks'. However, there are times, as we shall see, when self-modifying programs can come in useful.

Once again, the memory locations to which I shall refer are for the version you can buy from Norwich Computer Services with buffer%= &9FD4. If you have a different value then you can add, or delete, a few spaces between a line number, command or REM; or you can do the sums (in hex) instead. BASIC stores the value of its variables straight after the end of the program.

When I type in PRINT ~TOP I get &9F5B: this is where the variables start. If you RUN the program and press the up cursor a few times you will find a line at address &9F59 which contains the end-of-program marker (in hex) "0D FF". This is followed by the name of the first variable (without its first letter) essage$, and its value "Watch this space etc.". Move the cursor to the part of the value where the full stops are and type in a few characters. Because the message$ is printed every time a key is pressed (lines 780 or 1130) you will see the result of your key presses near the bottom of the screen immediately. Perhaps more amusing is to type in a hex code such as 0D which moves the printing position to the beginning of the line or 0A to print down one line. The value 0C (VDU 12) will clear the screen (a bit drastic?).

Is your mind clear ready for the next bit? The value of message$ is refreshed by the program at line 230 every time the loop, lines 220 to 250, is executed. If you enter an ASCII letter or hex code or if you press the left or right cursors then you will remain within FNmodify and so line 230 is not executed. That is why you can 'poke' the value of message$ and retain the modified value while it is printed by 780 or 1130. If you press <tab> or the up or down cursor then you exit FNmodify and return to the 220 to 250 loop where 230 refreshes the value of message$.

Now, if you poke the program, rather than the values of its store of variables, then you will make more permanent changes. In my version, line 230 is found soon after memory location &9264 (don't get mixed up with line 80). If you alter a full stop to, say, the letter M you will find that 4D appears in the hex code but nothing is printed in the "Watch this space etc." at the bottom of the screen. Now press <tab> to exit the FNmodify loop and you will see the M appear at the bottom of the screen. Press <esc> and LIST line 230. You will find that you have modified the program and that M has appeared in line 230. Be careful not to wreck the disc copy of the program: you have been warned!

For the Expert

Do you want to try something more ambitious? OK, then you can use the program to change line 1130 so that the 'PROC' becomes a 'REM'. The token (hex code) for PROC is F2 and that for REM is F4. RUN the program and use the cursor keys to move to memory location &9B43. Make sure you can recognise the location and (in hex mode) change the F2 to F4. Press <esc> and list line 1130. If you change line 790 (at &979D) similarly then these two changes will stop the program printing the message$ every time the value of message is 'poked'. Now go forward to &9F6F, where the value of message$ is stored, and change a few full stops: although the locations have been 'poked', the printing (lines 780 and 1130) has been disabled by changing PROC to REM, so the display at the bottom of the screen does not change as it did before.

So, you might ask, what use is this potentially dangerous facility to modify a program whilst it is still running? Bear with me – I've nearly got there. We can stay within the BASIC environment, within a program and use one part of the program to modify another part. Now consider the situation where the part we want to modify is written in assembly language mnemonics: we can run the BASIC compiler, look at the assembled code, call other Debugger functions (which is what this series is supposed to be about!) and even CALL the compiled program (provided it doesn't crash) all without leaving our original BASIC program.

Well, that's enough for this month. Those of you who want to steal a march on next month's sequel can try *LOADing a machine code program into the buffer and disassembling the code by using the debugger memory command `type$="I  "`. Try changing line 1370 to `PROCmemory("I  ",start%,&40,0,3)` as a start and I think you'll guess where we're going.

```
 10 REM > Memory01
 20 REM Author      :G L Fitton
 30 REM Copyright   :ABACUS TRAINING
 40 REM Version 1.00:11th March 1988
 50
 60 ON ERROR PROCerror
 70 MODE 3
 80 message$="Watch this space....
                ......................
                 ...................."
 90 buffersize%=&100 :REM Make sure
                      you have enough
100 DIM buffer% buffersize%
       :REM Reserve space in memory
110 start% = buffer% :REM Start of
                             display
120 pos% = 61:REM Position of cursor
130 vpos% = 11 :REM when altering
                     memory in ASCII
140 hpos% = (pos%-61)*3+11 :REM
                  Position of cursor
150 hvpos% = vpos% :REM when
            changing memory in hex
160 display% = FALSE :REM Redisplay
                  on screen when TRUE
170 ascii% = TRUE :REM When making
                    changes in ASCII
180 step% = 1 :REM Size of step
                     through memory.
190
200 PROChelp
210 REM *LOAD a file from disc to the
    buffer (see line 90 about size)
220 REPEAT
230   message$="Watch this space....
                ......................
                 ...................."
240   quit%=FNmodify :REM Modify
                    bytes in memory.
250 UNTIL quit%=TRUE
260 REM Save the modified file to
                               disc.
270 OSCLI("*SAVE buffer "+STR$~
   (buffer%)+"+"+STR$~(buffersize%)
                            +" 0 0")
280 *SETTYPE buffer &FFD
290 *STAMP buffer
300 CLS
310 PRINT "The file called""
            buffer"" now contains:-"
320 *DUMP buffer
330 PRINT
340 END
350
360 DEFPROCerror
370 *FX 4,0
380 CLS
390 REPORT
400 PRINT " at line ";ERL
410 ENDPROC
420
430 DEF PROCprint(message$,x%,y%)
440 REM Prints message at TAB(x%,y%)
450 LOCAL
460 PRINT TAB(x%,y%);
470 PRINT message$;
480 ENDPROC
490
500 DEF PROCmemory(type$,pstart%,
                   length%,x%,y%)
510 REM Displays a block of memory.
520 LOCAL command$
530 PRINT TAB(x%,y%);
540 command$="Memory"+type$+" "+
  STR$~(pstart%)+" +"+STR$~(length%)
550 OSCLI(command$)
560 ENDPROC
570
580 DEF FNalterASCII(RETURN pos%,
                       RETURN vpos%)
590 REM Lets user change the memory
600 LOCAL key%,key$
610 REPEAT
620   PRINT TAB(pos%,vpos%);
630   REM Cursor keys take CHR$ values
640   *FX 4,1
650   key%=GET
660   *FX 4,0
670   key$=CHR$(key%)
680   IF INKEY(-1)=TRUE THEN step%=8
                         ELSE step%=1
690   IF key%>31  AND key%<127 THEN
    PROCinsertASCII(key%,pos%,vpos%)
700   IF key%=136 THEN pos%=pos%-1
710   IF key%=137 THEN pos%=pos%+1
720   IF key%=138 THEN start%=start%
           +16*step%:display%=TRUE
730   IF key%=139 THEN start%=start%
           -16*step%:display%=TRUE
740   IF key%=  9 THEN ascii%=FALSE:
                       display%=TRUE
750   IF key%= 13 THEN quit% =TRUE
                      :display%=TRUE
760   IF pos%=60 THEN pos%=76:start%
           =start%-16:display%=TRUE
 770   IF pos%=77 THEN pos%=61:start%
           =start%+16:display%=TRUE
 780   PROCprint(message$,0,20)
 790 UNTIL display%=TRUE
 800 display%=FALSE
 810 =quit%
 820
 830 DEF PROCinsertASCII(key%,RETURN
                pos%,RETURN vpos%)
 840 REM Modifies the memory and
             prints to the screen.
 850 LOCAL key$,hex$
 860 key$=CHR$(key%)
 870 hex$=RIGHT$("00"+STR$~(key%),2)
 880 ?(start%+&80+pos%-61)=key%
 890 PRINT key$;TAB(8+(pos%-60)*3
                    ,vpos%);hex$;
 900 pos%=pos%+1
 910 ENDPROC
 920
 930 DEF FNchangehex(RETURN hpos%,
                   RETURN hvpos%)
 940 REM Lets user change the memory
 950 LOCAL key%,key$
 960 REPEAT
 970   PRINT TAB(hpos%,hvpos%);
 980   REM Cursor keys take CHR$ value
 990   *FX 4,1
1000   key%=GET
1010   *FX 4,0
1020   key$=CHR$(key%)
1030   IF INKEY(-1) THEN step%=8
                      ELSE step%=1
1040   IF INSTR("0123456789ABCDEF"
        ,key$) THEN PROCinserthex
             (key$,hpos%,hvpos%)
1050   IF key%=136 THENhpos%=hpos%-3
1060   IF key%=137 THENhpos%=hpos%+3
1070   IF key%=138 THEN start%=
        start%+16*step%:display%=TRUE
1080   IF key%=139 THEN start%=
        start%-16*step%:display%=TRUE
1090   IF key%=9 THEN ascii%=TRUE
                    :display%=TRUE
1100   IF key%= 13 THEN quit%=TRUE
                    :display%=TRUE
1110   IF hpos%= 8 THEN hpos%=56
     :start%=start%-16:display%=TRUE
1120   IF hpos%=59 THEN hpos%=11:
      start%=start%+16:display%=TRUE
1130   PROCprint(message$,0,20)
1140 UNTIL display%=TRUE
1150 display%=FALSE
1160 =quit%
1170
1180 DEF PROCinserthex(key1$,RETURN
                hpos%,RETURN hvpos%)
1190 REM Modifies mem'y,prints screen
1200 LOCAL key2$,hex$
1210 PRINT key1$;
1220 REPEAT
1230   key2$=GET$
1240   IF INSTR("0123456789ABCDEF"
              ,key2$)=0 THEN VDU 7
1250 UNTIL INSTR("0123456789ABCDEF"
                           ,key2$)
1260 PRINT key2$
1270 hex$=key1$+key2$
1280 hex%=EVAL("&"+hex$)
1290 IF hex%>31 AND hex%<127 THEN
      char$=CHR$(hex%) ELSE char$="."
1300 PRINT TAB(60+(hpos%-8)DIV3,
                      vpos%);char$;
1310 ?(start%+&80+(hpos%-11)DIV3)
                              =hex%
1320 hpos%=hpos%+3
1330 ENDPROC
1340
1350 DEF FNmodify
1360 PROCprint(message$,0,20)
1370 PROCmemory(" B",start%,&100,0,0)
1380 PROCprint(message$,0,20)
1390 IF ascii%=TRUE THEN
1400   pos%=(hpos%-11)DIV3+61
1410   quit%=FNalterASCII(pos%,vpos%)
1420   hpos%=(pos%-61)*3+11
1430   ELSE
1440   hpos%=(pos%-61)*3+11
1450   quit%=FNchangehex(hpos%
                          ,hvpos%)
1460   pos%=(hpos%-11)DIV3+61
1470 ENDIF
1480 =quit%
1490
1500 DEF PROChelp
1510 PRINT TAB(0,21);"Press cursors
          to move through memory."
1520 PRINT TAB(0,22);"Press <Shift>
       with cursors to move faster."
1530 PRINT TAB(0,23);"Press <Tab> to
      toggle between ASCII and hex."
1540 PRINT TAB(0,24);"Enter changes
      at cursor in ASCII or hex.  ";
1550 PRINT "Press return to finish.";
1560 ENDPROC
```

The Acorn I/O Podule

Brian Cowan

In one way, the I/O podule may be regarded as part of the plan of upward compatibility between BBC models; the Archimedes is very much a "BBC microcomputer". Thus the operating system Arthur is essentially a superset of the old B and Master OS. Similarly, the advanced disc filing system and the implementation of BASIC are upwardly compatible with those of the earlier machines. There are limits however. The Archimedes BASIC V assembler is for ARM code; programs using 6502 assembly language will not run. Help is at hand here with the 6502 emulator known as 65Arthur. Running this program the Archimedes emulates a BBC machine with a 6502 type processor. To be precise, the emulation is nearer that of a machine running a 65C102 second processor.

The I/O podule completes the exercise of imitation; it provides the input/output facilities found on the earlier BBC microcomputers. Thus there is a user port, a 1MHz bus and an analogue input, all having the usual connectors. The peripheral module or "podule" is the means of hardware interfacing to the Archimedes. There is firmware support for up to sixteen podules, but provision for only four podules inside the 400 series machines and two in the 300 series.

The podules plug into a vertical backplane mounted in the Archimedes. A four slot backplane is provided as standard on the 400 machines. This has two rows each with two side by side sockets. Thus up to four single width or two double width podules may be accommodated, or one double and two singles.

For the 300 series Acorn supply a two slot backplane to be purchased as an extra together with a cooling fan. The two sockets are mounted one above the other so that two podules may be fitted, either single or double width. (Solidisk should be providing a four slot backplane like that of the 400 for the 300 series machines)

The I/O podule is a double width unit. This is necessary because the back panel must accommodate the sockets for the user port, the 1MHz bus and the analogue port. There is also provision for a MIDI interface upgrade, and to this end there are two blanked off holes for DIN connectors on the panel, some empty i.c. sockets on the board together with plugs for a small extra board. The circuit board is thus somewhat wider than a single width board.

There is a 27128 16 kbyte ROM on the board. This contains the software for managing the podule. On powerup the contents of this ROM are read into the Archimedes memory and installed as a relocatable module.

The User Port

The user port hardware utilises the familiar 6502 family VIA chip, although here it is the cmos version, the 65C22. However, this is clocked at 2MHz unlike the 6522 VIAs in the earlier BBCs which run at 1MHz. While this doubling of speed will be advantageous in many applications, it does mean that wherever timing is important, programs will need modification.

Legal access to the user port is obtained using the OSBYTE calls 150 and 151 which communicate with SHEILA. As expected, it is the B side of the VIA with the buffered output which provides the user port. On previous BBCs the A side of the VIA was used for the parallel printer port however this is provided directly by the Archimedes. The A port is used for other purposes and is not accessed as part of SHEILA's area. Two lines of the A side, PA6 and PA7, are used for the "fire" buttons for the joystick/ paddle, connected through the analogue port. (On the previous machines these lines went to the system VIA.) One line, PA5, remains unconnected while the remainder, lines PA0 to PA4 are involved in paging the podule ROM.

It is not possible for the entire podule ROM to be addressed directly. The Archimedes interface to podules incorporates twelve address lines allowing access to 4k locations. And although sixteen data lines are supported, the I/O podule only utilises eight; the unit of data is thus the byte. Eleven address lines directly address the ROM and the remaining line enables the ROM output. The 16 kbyte ROM is thus read as eight 2 kbyte pages. (Actually the ROM contains only just over 1kbyte of code !) The pages are selected by lines PA0, PA1 and PA2 of the VIA. The remaining lines PA3 and PA4 go to the ROM PGM and Vpp pins.

The Analogue Port

The analogue port is based on the same twelve bit digital to analogue converter as in previous BBC machines, the µPD7002. And again the reference is provided by the voltage drop of three forward biased silicon diodes. Thus compatibility is guaranteed, even the lack of reference stability(!) Serious users of the analogue port may wish to remove the three diodes and replace them with a band gap reference. In fully compatible fashion, legal access to the analogue port is via the usual OSBYTE calls or, using ADVAL from BASIC.

The 1 MHz Bus

Notwithstanding the fact that the user port clocks at 2MHz and that the Master Compact implements a 2MHz bus (as does the Master through its cartridge sockets), the I/O podule provides a true 1MHz bus. The address lines and the data lines are buffered exactly as on the old BBCs and most other lines including JIM and FRED are provided. The exception is the interrupt lines IRQ and NMI. A jumper plug PL4 is provided on the podule board enabling these lines to be connected to either of the Archimedes interrupt lines IRQ (interrupt request) or FIQ (fast interrupt request). Also, pin 16 does not provide an analogue input to the machine's sound channel. Legal access is provided with OSBYTE 146 to 149 as expected.

Peek and Poke

Direct (although illegal) access to the podule facilities may be made since all I/O is memory mapped. However, this cannot be done from BASIC using indirection operators. The reason is that the podules sit in physical memory space and this cannot be accessed under normal operating conditions (ARM User Mode). Only logical addresses may be used, the translation being performed by the memory controller chip (MEMC). Physical memory, and hence the podule, may only be accessed in one of the privileged Supervisor Modes. This must be done using machine code.

It is possible, however, from within a BASIC program to assemble some code which takes the machine into SVC mode, reads and/or writes to physical addresses and then returns to User mode. Such direct access should rarely be necessary except where maximum speed is required. Even when operating under the 6502 emulator the old BBC addresses for SHEILA, JIM and FRED etc. do not apply since this is a "second processor" emulation; all 64 kbytes of memory refer to RAM. As with usual second processor operation, calls to the I/O devices using the OSBYTE calls are implemented correctly.

Each podule occupies a given address space in the Archimedes memory map. However, different types of podule and different types of data transfer are mapped to different areas. The

I/O podule is classified as a 'MEMC podule' and as such it connects directly to the memory controller (MEMC) chip of the machine. However some functions of the podule are really those of what is known as a 'simple podule'.

Simple podules connect to the input/output controller (IOC) chip which handles data transfer using one of four cycle types. Of these it is the 'synchronous transfer' that is used in the I/O podule. Simple podule (synchronous transfer) memory space maps to the podule ROM, the user port and the "fire" buttons. The 1MHz bus and the analogue to digital converter occupy MEMC podule space. The actual addresses are given in the I/O podule guide, although it is stressed that the absolute addresses may not apply to later machines.

Fortunately, the I/O podule software provides an instruction which returns the physical location of the I/O podule in the memory map. This is the software interrupt SWI "I/O_Podule_Hardware". The SYS instruction may be used from BASIC. Using this instruction, software can control a podule in any socket since the base address may be read in and all locations indexed with respect to this. The base address is in fact the first location of the simple podule (synchronous) space. This is the start of the I/O podule ROM. It is rather silly that the 1MHz bus, which is in MEMC podule space, sits below the base address. So negative indexing is required!

The MIDI interface circuitry that is on the main I/O podule board is essentially a serial interface circuit. There is a socket for a 6850 serial chip with transmit and receive lines. The input is isolated with a 6N138 opto-coupler. With suitable software this could have some useful applications without using the extra MIDI board. – A solution to the RS423 saga and the troublesome 6551 chips?

The 22 page guide which comes with the podule is just about adequate. There are descriptions of the various ports and mention of the incompatibilities with the old BBCs. The memory map is given together with a circuit diagram. A summary of the relevant OSBYTE calls is given, there are some example programs and a short index is provided.

Conclusion

The I/O podule does a good job in providing the usual BBC I/O facilities. Much of the circuitry is similar to the old BBCs and the podule provides the maximum possible in compatibility. Compatibility requires that even though the Archimedes is a 32 bit machine, and the podule data bus is 16 bits wide, the I/O podule operates on single bytes. For those not concerned with compatibility, or to supplement the Acorn I/O podule, there should be a third party 16 bit podule available soon. RESOURCE are planning a "versatile interface podule" (VIP) with full 16 bit input and output. This will also incorporate two eight bit 100 kHz analogue to digital converters.

Thanks to Tom Crane for help in preparing this review. A

More PRM Errors

• p 472. The name of the WIMP call at &400CE is "GetIconState" not "GetIconInfo"

User Guide Issue 2

– Additional Change

• p. 208 To enable the printer, use <ctrl-B>, not <ctrl-Break>! A

BBC to Archimedes Utility Programs

Matthew Treagus

The "BBC to ARM" utility disk (£15.95 from RESOURCE) contains several neat little conversion utilities including "BBCrun", a more complete version of 65Arthur as well as UDG's (User Defined Graphics) Conversions, Picture Conversions and a Conversion Aid.

BBCrun – The Emulator

BBCrun is some 4k longer than 65Arthur and is installed as a module and called with *BBCrun. The emulation is not of a 6502 Second Processor environment but of a BBC 'B' I/O processor. The following additional emulations have been made since 65Arthur.

*FX 234 – detect TUBE

*FX 154 – Video ULA (partial emulation)

– Only turns cursor on and off

VDU queue – Updates HIMEM after MODE changes

Screen files – Screens can be *LOADed and *SAVEed in Modes 0, 1, 2, 4, 5 and 7.

UDG chars – Definitions are now stored in locations &C00-&CFF :

&367-&37F – Locations emulated that contain font details and palette settings

But such things as sideways RAM and ROM are not possible and the screen is still not memory mapped, so screen dumps and games software are still not possible. The break key and SOUND and ENVELOPE keywords are not emulated, nor are certain operating system vectors, e.g. the event vector. Other Master series features *SHADOW and modes > 7 are not emulated.

DFS Emulation

The DFS emulator within the BBCrun emulator allows ADFS directories to act as DFS drives. The directories (drives) of programs must first be "flattened" to set up the internal DFS directories that are attached to the ADFS filenames, this is done using the 'Makedisk' utility. Then the LINKs must be setup. i.e.

```
*LINK 0 DR0
*LINK 1 Pics
```

Sets DFS drive 0 to be represented by ADFS directory 'DR0' and drive 1 to be 'Pics'. The *DRIVE n command is then used to change the drive, temporary access using :n is also possible and *LIB drives can also be used. OSARGS and OSGBPB are also supported to an extent and the catalogue is maintained at &E00 so in fact the DFS workspace is also emulated to a degree. Thus, certain 'naughty' packages can be run.

The emulator works very well but unfortunately is only a partial emulation. *INFO for example is unsupported as is *TITLE these commands still perform their ADFS functions. Once a disk/dir has been converted for use with the BBCrun it makes a bit of a mess of the filenames and conversion back to ADFS is not an easy job – **you have been warned!**

BBCscan

BBCscan simply asks for a BASIC filename and then proceeds to search the BASIC file for any possible problems. These are then displayed on screen or sent to a printer. A line number, message reference and message text are returned at problem lines. The message reference can then be looked up in the manual for hints on conversion. e.g. 101 / 32 / Softkeys

This means that at line 101 reference 32 there is a problem with *SAVEing softkeys. The manual provided the comment "needs conversion" but other comments were slightly more useful!

Picconv

This simply converts BBC 6502 *SAVED screens into sprite files for *SCREEN LOADing. MODE7 files need to be *PRINTed from the original program. Picconv also allows conversion of GXR sprites created using the Acorn GXR ROM or the Master into Archimedes Sprites.

UdgConv

Converts character definitions into VDU 23 statements as a file that can be APPENDed to BASIC programs or *EXECed that set up the characters in the native ARM mode.

Conclusion

The 'BBC to ARM' package is a major step forward for BBC emulation. It is by no means a complete emulation (although a touch better than the Master/BBC compatibility, well almost!) I would like to see a complete BBC system emulation and I would have thought one would not be too far away hopefully containing:

OSWORD &7F emulation, Fully memory mapped screen, 16 Socket Sideways Ram including service ROMs and possibly certain add-ons like a 2nd Processor and Shadow Screens and then ultimately all of this on a ROM with possibly the 6502 chip and all this on a network too. (I don't ask much!)

This is very useful program for anyone wanting to convert BBC programs but especially so for inexperienced programmers and schools who do not have the knowledge or the time to spend converting programs. If you are going to be doing any conversions then my advice would be to go out and buy it straight away.

For further information try the Acorn Applications Note "Archimedes 6502 Emulation 0340011" from Customer Support Dept. at Acorn's Fulbourn Road address.

"£2.50/note, subject to availability"

DROOM!

Tim Beverley

The aim of the game is to help Henry the frog to get out of the spell which the dreaded dragon, Droom cast upon him. You see, Henry is not really a frog – he is a prince, and if you can help him he will be able to turn back into a prince.

You must seek out the dreaded dragon, Droom. But that is not all, he also carried Princess Arminda away and has chained her in his dungeon. If you can free her, she will change Henry back into a prince by kissing on his little green cheek. Little Bit will help you too. You will need to visit the Wizard. He is always happy to see you close to tea-time. The Witch hates the dragon and she will give you all the help she can. The Fairies are your friends. Droom eats fairies for breakfast. Everyone hates Droom.

My favourite game is Crossing the Floor; you have to find a path which is cool because if you go onto a hot square you go back to the start and every time you do the game there is a different pattern. I think the program is really GREAT.

Dad's comments

Droom is not just a game (or rather a series of games) it is a suite of educational material which can be used to support the development of mathematical concepts – binary, xy-co-ord-inates, pattern matching – and to aid logical thinking and memorising sequences. Sounds really **boring**, doesn't it!?! It's not though – it's a fun game for all the family. When Sue gets on it, I can't get my Archimedes back even if I say I need it for work!

For £18.95 + VAT you get the main DROOM disc (which, on its own, keeps my two quiet, or I should say occupied, for hours) plus a disc of supplementary exercises on each of the different topics, a manual to tell the teacher how to use Droom to the best advantage in the classroom and a book which gives the kids "The story so far...". If you think education should be fun then Droom is for you.

Archimedes 5.25 Disk Interface.

Fit a 5.25 Drive to your Machine the safe and easy way using our
NEW DISK INTERFACE

- Available Now
- No Soldering
- Supports 4 Drives
- Low Power Consumption
- Fully Buffered
- Easy to Fit
- All Cables Supplied
- Full Instructions

In Stock Now and Despatched the same day on receipt of your
Cheque / Postal Order for £27.95 inc P & P

Dudley Micro Services

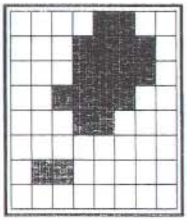
30 Hadley Close, Netherton, Dudley, West Mids. DY2 9JX

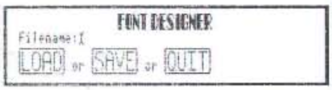
Tel: (0384) 633142

Font Designer

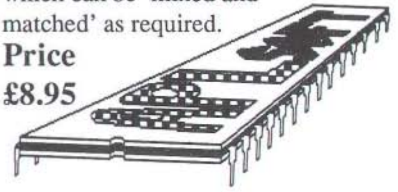
Design your own 8x8 BBC-fonts for the Archimedes. Fully mouse driven software. Price includes comprehensive manual and 15 different sample fonts, including scientific symbols, all of which can be 'mixed and matched' as required.

Price
£8.95





FONT DESIGNER
Filename:I
LOAD or SAVE or QUIT



To: APL Software, 7 Ascendale, Deeping St James, Peterborough, PE6 8NZ.
Please supply one copy of **Font Designer**. I enclose a cheque/postal order for £8.95

Name:...

Address:..

....

Which ROM Podule?

Having tried out both the two currently available ROM podules, here are our findings…

Capacity

Both podules have 8 ROM slots one of which is taken up with a manager ROM (and you need one manager per board even if you put in multiple ROM boards). Computer Concepts' podule can have RAM or ROM in any of the remaining 7 slots whereas on the Acorn board, only two of the sockets can have RAM in them and they only mention 32k RAM chips whereas Computer Concepts say you can put 128k static RAMs in when they get down to a sensible price.

Ease of setting up

In terms of setting up the hardware, the Computer Concepts board has one link per ROM to decide whether the battery backup supply should be applied to it whereas the Acorn board has two links for each ROM which have to be set in order to determine what size of ROM or ROM is to be used in that socket. Then to set up the software, with the Acorn board you have to do a "*Configure ROMboard" for each slot to specify what type of ROM or RAM it contains whereas the Computer Concepts board has a *AUTOCONFIGURE command which looks at each slot in turn, finds out what is in it and configures it accordingly.

Mutual compatibility

If you have both a Computer Concepts board and an Acorn board in the machine, you will find that the Computer Concepts software (*RFS) allows you to access the Acorn board, but that the Acorn software (*ROM) does not recognise that the Computer Concepts ROM board exists.

Data transfer speed

The Computer Concepts board can load and save at around 300 kbytes per second whereas the Acorn board is more like 30 kbytes per second which is only about twice the speed of the ADFS 3.5" disc. Unfortunately even if you access the Acorn board through the RFS software of the Computer Concepts board, the access time is no better. (Computer Concepts quoted 300 kbytes per second for their board and 75 kbytes per second for the Acorn board, but I timed it at about 30.)

These timings are not exact because a 'feature' of the podule manager on the 1.2 OS is that it doesn't update the system clock during access to the podules so that you cannot use TIME, TIME$ or *TIME. It doesn't actually stop the hardware clock, but to get *TIME and TIME$ to catch up with the hardware clock you have to do a <ctrl-break>. So if you are using an application which does a lot of podule access you will find that the clock runs slow! I guess that this applies to the I/O board as well, but since you are not accessing large chunks of information at a time, the effect is probably too small to matter. The same problem affects the sound system. If you want some very funny sounds, try:

```
VDU7 : REM make a bleep
FOR N%=1TO10
*LOAD ROM:filename 10000
NEXT
```

Battery backup

You can buy the Computer Concepts board with battery backup installed but although the Acorn board has space for the battery, all they do is tell you the parts needed and how to fit it, including what kind of soldering iron to use! They do also say that "your Acorn dealer should be able to assist you in obtaining these items". They also say, of course, that fitting the battery backup invalidates your guarantee!

BASIC Cross-Referencer

Brian Carroll

The idea of this program is to provide you with a list of the names of all the procedures, functions and variables you have used and the numbers of all the lines on which they are referrenced. The program also provides an optional printout of the information.

Setting up

First of all, alter the printer set-up procedure at the end of the program to suit your printer and save the modified program to disc.

The program asks for the filename of the program to be tested and it loads the program into a buffer. For large programs, you may need to increase the size of the buffer (line 170). If the buffer is too small, you will get an 'Address exception' error. (You may also need to increase the size of reflist% at line 180.)

```
 10 REM >XREF
 20 REM *************************
 30 :
 40 REM Copyright: J.A.Hall 1984,
                 B.Carroll 1984 & 1988.
 50 :
 60 REM Originally written for BBC
    micro; now improved & adapted for
     Archimedes by B.Carroll; Feb 88.
          Prestel Mailbox: 025222539.
 70 :
 80 REM   XREF reads a BASIC file
    from disc and produces a sorted
       list of all the PROCs, FNs &
        variables and the line nos.
                of each occurrence.
 90 :
100 REM   Acknowledgement is due to
 THE BBC MIC[illegible]GUIDE by G.Blackwell
      for tree-handling PROCs 'do',
      'left','right' and 'compare'.
110 :
120 REM Alter codes in PROCset_
        printer (at the end) to suit
                       your printer.
130 :
140 REM ***************************
150 MODE 7:save_at%=@%
160 PROCset_on_error
170 bufsize%=1024*24:REM Space for
                       program (24K)
180 DIM buffer% bufsize%,reflist%
                           (1999,1)
190 DIM name$(999),nr%(999),
     nl%(999),refp%(999),curr%(999)
200 :
210 PROCfrontpage
220 PROCclear
230 :
240 PRINTTAB(0,2)CHR$(131)"Insert
        new disc now, if necessary"
250 PROCclear
```

(Continued form opposite page...)

Availability

Here's the problem. At the moment the Computer Concepts ROM podules are in very short supply whereas the Acorn podules are "off-the-shelf".

Cost

The VAT inclusive prices, through Archive, are: Acorn's board (no battery backup) £63, Computer Concepts's board £53 without and £63 with battery backup.

Conclusion

Sorry, Acorn, but the facts speak for themselves! The only reason I can see for anyone buying the Acorn board is if they have to have one NOW and cannot wait for supplies of the Computer Concepts board.

(We are accepting orders for Computer Concepts ROM podules on the basis that we hold the cheques but do not bank them and supply the podules in strict rotation as we receive them from C.C.)

```
260 *MOUNT 0
270 *DIR $
280 PRINTTAB(0,2)CHR$(131)"Enter
               FULL path & file name"
290 ON:PRINTTAB(3,4)">>> ";:
                INPUTLINE ""file$:OFF
300 PROCload(file$)
310 CLS
320 :
330 PROCanalyse
340 :
350 PRINTTAB(0,16)CHR$(132)CHR$(157)
                           CHR$(131);
360 PRINT" Do you want hard copy
                              (Y/N)? ";
370 G%=0:*FX15,1
380 ON:REPEAT:G%=(GET AND &DF):UNTIL
    G%=89 OR G%=78:CLS:PROCsound:OFF
390 IF G%=89 AND FNtest_printer
       =FALSE PROCwarn("PRINTER NOT
                   READY!"):GOTO350
400 *FX21,3
410 IF G%=89 PRINTTAB(0,16)CHR$(132)
                 CHR$(157)CHR$(131);
420 IF G%=89 PRINTSPC10"P r i n t
                   i n g";SPC10;:VDU2
430 IF G%=78 PRINTTAB(0,16)CHR$(132)
                 CHR$(157)CHR$(131);
440 IF G%=78 PRINT" Press SHIFT to
         scroll screen     ";:VDU14
450 VDU28,0,21,39,7
460 PROCset_printer
470 PROCprint_hdg
480 :
490 @%=6:PROCoutput(0)
500 IF G%=89 VDU1,13,1,13:FOR Z%=0
      TO 74:VDU1,95,:NEXT:VDU1,12,3
510 VDU28,0,23,39,7:REM Change window
520 :
530 PRINTTAB(0,16)CHR$(132)CHR$(157)
                           CHR$(131);
540 PRINT" Do you want to repeat
                (Y/N)?  ";:PROCsound
550 G%=0:*FX15,1
560 ON:REPEAT:G%=(GET AND &DF):UNTIL
       G%=89 OR G%=78:PROCsound:OFF
570 IF G%=89 VDU15:GOTO350
580 PRINTTAB(0,16)CHR$(132)CHR$
   (157)CHR$(131)"          Finished"
                              ;SPC15;
590 :
600 VDU1,27,1,64,3,15,26: REM Exit
                       point, tidy up
```

```
610 ON:@%=save_at%:CLOSE#0:
                         ON ERROR OFF
620 END
630 :
640 REM ***************************
650 :
660 DEFPROCanalyse
670 namecount%=0:linecount%=0:
                        linetotal%=-1
680 K%=-1:@%=5
690 nr%(0)=-1:nl%(0)=-1:lineNo%=-1
700 PRINTTAB(0,12)CHR$(132)
                 CHR$(157)CHR$(131);
710 PRINT"  Processing lines
     ..........."CHR$(156)CHR$(129)
                   CHR$(136)"WAIT!"
720 VDU 28,3,15,32,9
730 byte%=FNfetch
740 IF byte%<>13 THEN PRINT"Missing
                 'START LINE'":GOTO730
750 ass%=FALSE
760 :
770 REPEAT:REM Outer loop. Read
        each line of BASIC program
                     being examined
780  linetotal%+=1:REM Count lines
                            examined
790  hibyte%=FNfetch
800  IF hibyte%>254 GOTO1370:
                REM EXIT from outer
                      loop, 'cos ..
810  lobyte%=FNfetch:REM ... 'TOP'
                    (&FF=255) found
820  lineNo%=hibyte%*256+lobyte%
830  PRINT lineNo%;
840  linelen%=FNfetch
850  IFlinelen%=4 GOTO1350:REM
            Empty line, so skip it
860  inquotes%=FALSE:ignore%=FALSE
870  var_ident%=FALSE:hex%=FALSE
880  start%=TRUE:bytecount%=0
890  proc%=FALSE:funct%=FALSE
900  :
910  REPEAT:REM Inner loop. Read
                      rest of line
920   bytecount%+=1
930   byte%=FNfetch
940   IF bytecount%=linelen%-3
      AND byte%<>13 PRINTCHR$(129)
          "Missing <CR>":GOTO 930
950   IF start% AND byte%=32 GOTO
      1330:REM Ignore initial spaces
```

```
 960   IF start%ANDbyte%=42 ignore%
           =TRUE:REM Ignore OS calls
 970   IF byte%=&F4 ignore%=TRUE:
              REM Token &F4 = 'REM'
 980   IF byte%=&DC ignore%=TRUE:
             REM Token &DC = 'DATA'
 990   IF ignore%  GOTO1330:REM
               Skip if 'ignore' set
1000   IF byte%=34  inquotes%
                   =(NOT inquotes%)
1010   IF inquotes% GOTO1330:REM
                Skip literal string
1020   IF byte%=&F2 proc%=TRUE :
             REM Token &F2 = 'PROC'
1030   IF byte%=&A4 funct%=TRUE:
         REM Token &A4 = 'FunctioN'
1040   IF byte%=58  start%=TRUE:REM
         Re-start at colon separator
1050   IF byte%=91 ass%=TRUE:REM
                 [ = start assembler
1060   IF byte%=93 ass%=FALSE:REM
                   ] = end assembler
1070   IF ass% AND NOT var_ident%
               GOTO 1330: REM Skip
               assembler mnemonics
1080   IF var_ident% GOTO1230:REM
             JUMP; start of a label
1090   IF byte%=38  hex%=TRUE:GOTO
           1330:REM &=hex identifier
1100   IF hex% AND (byte%<48 OR
         (byte%>57 AND byte%<65) OR
              byte%>70) hex%=FALSE:
           REM Not a valid hex digit
1110   IF byte%<>&8D GOTO1140:REM
            &8D not a token, so ...
1120   FOR Z%=1 TO 3:byte%=FNfetch
                              :NEXT
1130   bytecount%+=3:GOTO1330:REM
                ... discard 3 bytes
1140   IF byte%<64 OR (byte%>90 AND
            byte%<95) OR byte%>122
              GOTO1330:REM Invalid
                variable character
1150   IFhex% GOTO1330:REM Hex number
                          starting
1160   :
1170   var_ident%=TRUE:REM So
            what's left must be a…
1180   label$=CHR$(byte%):REM ...
              var, PROC or FN name
1190   IF proc% label$="*PR "+
         label$:proc%=FALSE:REM Add
               ident for PROC name
1200   IF funct% label$="*FN "+
         label$:funct%=FALSE: REM Add
                   ident for FN name
1210   GOTO 1330
1220   :
1230   IF byte%=36 OR byte%=37 PROC
            build:REM Add var-type
                   ident, % or $
1240   IF (byte%>64 AND byte%<91) OR
         (byte%>94 AND byte%<123) OR
            (byte%>47 AND byte%<58)
                PROCbuild:GOTO1330
1250   var_ident%=FALSE:REM End of
            label if not A-Z, a-z,
                       0-9, _ or #
1260   :
1270   REM Search name table to see
                  if already there
1280   match%=FALSE
1290   IF namecount%=0 name$(0)=
         label$ ELSE PROCcompare(0)
               :REM Add No. to list
1300   IF NOT match%  np%=namecount%
1310   PROCadd(lineNo%)
1320   IF NOT match%  namecount%+=1
1330  UNTIL bytecount%=linelen%-3:
       REM i.e. until whole of line
                            checked
1340   :
1350 UNTIL FALSE
1360 :
1370 VDU28,0,23,39,7
1380 PRINTTAB(0,12)CHR$(132)
                CHR$(157)CHR$(131);
1390 PRINT" Number of lines read is"
                CHR$(129)linetotal%
                "      ":PROCsound
1400 PROCclear
1410 ENDPROC
1420 :
1430 DEFPROCbuild
1440 label$+=CHR$(byte%)
1450 ENDPROC
1460 :
1470 DEFPROCcompare(N%)
1480 IF label$=name$(N%) np%=N%:
                match%=TRUE:ENDPROC
1490 IF label$<name$(N%) PROCleft(N%)
                  ELSE PROCright(N%)
1500 ENDPROC
1510 :
```

```
1520 DEFPROCleft(N%)
1530 IF nl%(N%)=-1 nl%(N%)=namecount%
        :name$(namecount%)=label$:nl%
  (namecount%)=-1:nr%(namecount%)=-1
             ELSE PROCcompare(nl%(N%))
1540 ENDPROC
1550 :
1560 DEFPROCright(N%)
1570 IF nr%(N%)=-1 nr%(N%)=namecount%
        :name$(namecount%)=label$:nl%
  (namecount%)=-1:nr%(namecount%)=-1
             ELSE PROCcompare(nr%(N%))
1580 ENDPROC
1590 :
1600 DEFPROCadd(lineNo%)
1610 reflist%(linecount%,0)=lineNo%
1620 reflist%(linecount%,1)=-1
1630 IF match%  reflist%(curr%(np%)
       ,1)=linecount% ELSE refp%(np%)
                         =linecount%
1640 curr%(np%)=linecount%
1650 linecount%+=1
1660 ENDPROC
1670 :
1680 DEFPROCprint_hdg
1690 VDU1,14,1,27,1,45,1,1:REM Double
                  width, underline on
1700 PRINT"CROSS-REFERENCE LIST"'
1710 PRINT TIME$;".  Path/file name:
                               "file$
1720 VDU1,27,1,45,1,0:REM U'line off
1730 PRINT'
1740 VDU1,27,1,45,1,1:REM U'line on
1750 PRINT"FN/PROC/VAR";
1760 VDU1,27,1,45,1,0:REM U'line off
1770 PRINTSPC6;
1780 VDU1,27,1,45,1,1:REM U'line on
1790 PRINT"LINE NUMBERS"
1800 VDU1,27,1,45,1,0:REM U'line off
1810 ENDPROC
1820 :
1830 DEFPROCoutput(N%)
1840 IF nl%(N%)<>-1  PROCoutput(nl%
                                (N%))
1850 PROCprint_table(N%)
1860 IF nr%(N%)<>-1  PROCoutput(nr%
                                (N%))
1870 ENDPROC
1880 :
1890 DEFPROCprint_table(N%)
1900 len%=LEN(name$(N%))
1910 IF len%>15 THEN name$(N%)=LEFT$
           (name$(N%),14)+">":len%=15
1920 PRINT'name$(N%);SPC(15-len%);
1930 next%=refp%(N%)
1940 num%=0
1950 REPEAT
1960   lineNo%=reflist%(next%,0)
1970   next%=reflist%(next%,1)
1980   PRINT lineNo%;
1990   num%=num%+1
2000   IF num% MOD 10=0 AND next%<>
                  -1 PRINT'SPC(15);
2010 UNTIL next%=-1
2020 ENDPROC
2030 :
2040 DEFFNfetch:K%=K%+1:=buffer%?K%
2050 :
2060 DEFPROCload(f$)
2070 LOCAL L%,:L%=0
2080 chan%=OPENIN(f$)
2090 PRINTTAB(0,12)CHR$(132)CHR$
                     (157)CHR$(131);
2100 PRINT" Reading file to buffer
            ...."CHR$(156)CHR$(129)
                  CHR$(136)"WAIT!"
2110 WHILE NOT EOF#chan%
2120   buffer%?L%=BGET#chan%
2130   L%+=1
2140 ENDWHILE
2150 CLOSE#chan%
2160 ENDPROC
2170 :
2180 DEFPROCfrontpage
2190 title$="CROSS REFERENCE for the
                         Archimedes"
2200 auth$=" (C)1984, John Hall &
                     Brian Carroll"
2210 PRINTTAB(0,1)CHR$(132)CHR$(157)
2220 PRINTTAB(0,2)CHR$(131)CHR$(157)
          CHR$(132)CHR$(141)title$
2230 PRINTTAB(0,3)CHR$(131)CHR$(157)
          CHR$(132)CHR$(141)title$
2240 PRINTTAB(0,4)CHR$(132)CHR$(157)
2250 PRINTTAB(0,5)CHR$(132)CHR$(157)
                     CHR$(131)auth$
2260 PRINTTAB(0,6)CHR$(132)CHR$(157)
2270 VDU28,0,23,39,7
2280 PRINTTAB(0,2)CHR$(130)"This
  utility provides a listing of all"
2290 PRINTCHR$(130)"the variables,
          procedures and functions";
2300 PRINTCHR$(130)"in any BASIC
          program (read as a file),"
2310 PRINTCHR$(130)"together with
            the line number of each"
2320 PRINTCHR$(130)"occurrence."
```

```
2330 PRINTTAB(0,8)CHR$(130)"Names are
       listed in alphabetical order"
2340 PRINTCHR$(130)" (upper case
           first, then lower case)."
2350 PRINTTAB(0,11)CHR$(130)"Names
      longer than 10 characters are"
2360 PRINTCHR$(130)"truncated and
                   marked with '>'."
2370 ENDPROC
2380 :
2390 DEFPROCsound:SOUND1,-1,100,1
                            :ENDPROC
2400 :
2410 DEFPROCset_on_error
2420 ON ERROR IF ERR=17 GOTO600 ELSE
    IF ERR=&D6 OR ERR=&DE OR ERR=&CC
            GOTO2440 ELSE GOTO2450
2430 ENDPROC
2440 CLS:VDU7:PROCwarn("FILE NAME
       PROBLEM!"):PROCclear:GOTO240
2450 VDU7,12,26:@%=0:REPORT:PRINT" @
                   line "ERL:GOTO600
2460 :
2470 DEFPROCclear
2480 *FX 15,1
2490 ON:PRINTTAB(0,16)CHR$(132)CHR$
                    (157)CHR$(131);
2500 PRINT" Press any key to
  continue ";:G%=GET:VDU12:PROCsound
2510 OFF:ENDPROC
2520 :
2530 DEFPROCwarn(m$)
2540 PRINTTAB(7,5)CHR$(141)CHR$(136)m$
2550 PRINTTAB(7,6)CHR$(141)CHR$(136)
                            m$:VDU7
2560 ENDPROC
2570 :
2580 DEFFNtest_printer
2590 *FX21,3
2600 VDU2,1,0,1,0,1,0,1,0,1,0,1,0,
                              1,0,3
2610 TIME=0:REPEAT UNTIL TIME>30
2620 IF ADVAL(-4)=63 =TRUE ELSE
                             =FALSE
2630 :
2640 DEFPROCset_printer:REM Set up
                  for CANON PW1080A
2650 VDU1,27,1,64,1,27,1,40:REM
                         Reset, NLQ
2660 VDU1,27,1,108,1,5,1,27,1,78,1
       ,10:REM LM 5, perf. skip 10
2670 ENDPROC
```

More about Brainsoft

Mark Sealey

Archive has in the past looked at various methods of transferring files from 5.25" format to the Archimedes ADFS 3.5" disks. One such is the Transfer Software supplied with a lead from Brainsoft at £14 inclusive of VAT and p & p. An earlier release was indeed reviewed in Archive Vol 1 number 3.

Since then Brainsoft has sent some five or six disks each slightly better than its predecessor. A final version has been evaluated for this review, found to work well and to contain some additional features. This update can be obtained by sending back older disks to Brainsoft with an address label and 56p worth of stamps. One of the things you get when you do is an enlarged though still rather scrappy manual.

It now does work – with no need for patches – with the 1.2 Operating System. It is possible to transfer much larger files than previously; indeed there is claimed to be no limit to their size. It is also possible to configure the suite for acceptable error rates. I found that it worked well when set for zero errors. Most importantly, perhaps, is that it will work with both DFS and ADFS format at the BBC end. The suite will automatically attempt to set the correct Filetype for a transferred program's new environment.

Into the bargain, Brainsoft have produced an Editor for use by those wishing to write source code in any of the batch of compiled languages produced so far by Acorn. An example in Fortran comes with the program, which is booted from Brainsoft's own version of the Desktop. It has one or two pleasing features being mouse and icon-controlled for most of the main functions. Block manipulation is also achieved by the three mouse buttons.

It does not claim to be highly sophisticated but can work with two text files open at once and in separate windows. There is search and replace and merge as well as an indication of space used so far and a facility to number your code for easier debugging.

Brainsoft have here a product that is possibly overdesigned; a fully blown wimp interface for something that really does relatively little. Yet if you want an alternative to TWIN at under a third of the price with a maximum of 5000 lines capacity (though it is claimed that files can be linked: I did not try this), it's well worth considering. You would be well advised to exit from the suite legally as it temporarily configures the Archimedes into all sorts of nasty options. The Editor without the Transfer Utility Lead costs £13 and with the lead £19.

Program Disc Subscriptions?

We have been asked if we will do subscriptions for the program discs but I don't really think that, with the work involved, it would be possible to reduce the cost. What we can do though is to allow you to order several discs at a time, i.e. including discs not yet published. What we would then do would be to prepare the sticky address labels and mark them all with the issue numbers to be sent out. You would just have to keep a note of which ones you had ordered so that you knew when to re-order.

The other idea we are looking into is setting up an Archive Bulletin Board from which, once you have paid the subscription, you would be able to down-load the programs free of charge.

Screen-Writing

Adrian Look *(I squeezed this article in at the last minute! Sorry, Adrian!)*

Writing directly to the screen on other Acorn machines was frowned upon, although many programs did so (mainly games). This was done for extra speed in processing graphics or to create special effects. As a result, the programs lost their compatibility. Acorn have decided, however, that they would support direct screen accessing on the Archimedes(!) but the programmer must follow certain ground rules. If you don't do so then your programs may not work on subsequent machines. The 'rules' that should be followed are more of a help to the programmer than a hindrance. Basically, all the programmer has to do is use the screen data provided by an OS call, rather than assuming it. That way his routines will always find the screen location and format – no matter what!

'OS_ReadVduVariables' – SWI &31

This call allows you to access all sorts of useful variables, in any order you wish. Each variable has a number assigned to it, all you have to do is produce a list of variable numbers and the call will supply the corresponding values (see the example program). Here is a list of the variables you can access:

Var no:	Value and meaning
	Screen mode information (this is the important stuff)

0	bit0 =0 graphics mode
	=1 non-graphics mode
	bit1 =0 non-teletext mode
	=1 teletext mode
	bit2 =0 non-gap mode
	=1 gap mode (i.e. gaps between text)

1	number of text columns minus one
2	number of text rows minus one
3	number of colours available minus one

4	number of pixels	=0	1280 pixels
		=1	640 pixels
		=2	320 pixels
		=3	160 pixels
148	screen start address used by VDU drivers		
149	screen start address used by hardware		
150	size of configured screen memory		
161	highest mode available		

Graphics window co-ordinates (in)

128	left x
129	bottom y
130	right x
131	top y

Text window co-ordinates (in)

132	left x
133	bottom y
134	right x
135	top y

Graphics origin (ex)

136	x co-ordinate
137	y co-ordinate

Graphics cursor position (ex)

138	x co-ord
139	y co-ord

Graphics cursor position (in)

144	x co-ord
145	y co-ord

Graphics colour information

151	GCOL action for foreground colour
152	GCOL action for background colour
153	foreground colour
154	background colour
157	foreground tint colour
158	background tint colour

Text colour information

155	foreground colour
156	background colour
159	foreground tint colour
160	background tint colour

```
 10 REM >$.ScrnData
 20 :
 30 REM **************************
 40 REM *  Reading VDU variables  *
 50 REM * written by  Adrian Look *
 60 REM *      27th March 1988     *
 70 REM **************************
 80 :
 90 DIM block_in%  &400
100 DIM block_out% &400
110 offset=0
120 :
130 var_no=0
140 WHILE var_no<>-1
150   INPUT "Which variable (-1 to
                  terminate):"var_no
160   block_in%!offset=var_no
170   offset+=4
180 ENDWHILE
190 :
200 SYS "OS_ReadVduVariables",
                block_in%,block_out%
210 :
220 FOR value=0 TO offset-8 STEP 4
230   var_val=block_out%!value
240   PRINTvar_val
250 NEXT
260 END
```

'OS_ReadModeVariables' – SWI &35

This call will allow you to read the variables 0-4 of any mode (as listed above), without changing mode. For example:

```
mode=12
var_no=1
SYS "OS_ReadModeVariables",
   mode,var_no TO ,,var_value
PRINT var_value
```

Next month, I shall describe some more SWI calls related to the graphics. These will include reading and setting the palette in a 256-colour mode – complete with editor.

Small Ad's

If you would like to insert small ad's, free of charge, send them to us. Each should be less than 30 (thirty) words long and should relate to **Archimedes** and associated devices, i.e. no adverts for old BBC B's and Masters!

*HELP Archive

What are its aims? – Archive is a subscription magazine for users of the Acorn Archimedes range of computers, which aims (1) to provide information to the user (2) to provide a forum in which we can all share ideas (3) to give the benefit of bulk buying of software (4) to allow software and hardware vendors to advertise their wares.

Is it a User Group? – No, but I would like it to have a "User Group feel" – like BEEBUG when it first started – except that Norwich Computer Services has to earn a living from the magazine – hence the order form overleaf.

However, Sue and I are not in this business to make lots of money; we enjoy the work we do and it's very satisfying to be able to provide a useful service. We only ask to make enough money to live on plus a bit to give away and then we'll be happy. (I hope it doesn't sound too trite, because it's true.)

HELP! – Could you help us, please, to make Archive a success? We don't have the muscle or the financial budget of the big magazines and cannot afford to do a vast amount of advertising, so, if you think Archive is good, please take out a full subscription (if you have not already done so) and recommend the magazine to a friend or two. If you want any more information sheets and subscription forms, please let us know.

Can we answer technical enquiries by phone?

As much as we would like to continue the policy we have always had of being available to answer all your technical enquiries by phone, we felt that if we were not careful, we would be so inundated with calls that we would not have time to get the information out to you through the magazine. For that reason, we have introduced the Technical Help Service so that those who really need the instant access to help can purchase it. (£8 per year.) I hope you will bear with us and, if you do not feel it is worth the extra £8, please send your enquiries by post.

Do we take Access or Visa? No, I'm afraid not – mainly because of the high percentage that we would have to pay for the privilege (I think it's 6% when you first start). The other reason is that we have always had a policy of sending goods out by return of post, whenever possible, regardless of whether the cheques have cleared through the bank. The way we see it is that if you trust us by sending a cheque without the Access guarantee, it is reasonable for us to trust you by sending out the goods without waiting for the cheque to clear. Perhaps we are foolish to take the risk, but we have only had two dud cheques in three years of full time trading.

Those who help us... Although there's only two of us here, we are surrounded by a lot of people who help us in various ways, some voluntarily and some professionally, without whom we would not be able to stay in business. I don't think it is appropriate to mention them all by name, but I would just like to assure them that we are really grateful to them all. However, as I have said in most of my previous publications, as a committed Christian, there is One Person who never fails us and without whom we would achieve nothing worthwhile. God supplies all our needs, and we thank and praise Him!

I hope you find Archive interesting and informative and that you will feel able to contribute your own ideas, hints and tips or even full articles. I'm afraid that we can't afford to pay vast sums for articles, but at least you'd automatically become an "HLMTHS". Honorary Life Member, Technical Help Service!

Thanks again,

Paul Beverley

Fact-File

4Mation	Linden Lea, Rock Park, Barnstaple, EX33 9AQ. (0271 – 45566)
ACE Computing	27 Victoria Road, Cambridge, CB4 3BW. (0223 – 322559)
APL Software	7 Ascendale, Deeping St James, Peterborough, PE6 8NZ.
Brainsoft	22 Baker Street, London, W1M 1DF. (01 – 486 – 0321)
CCD Computer Services	71 Marlborough Park Avenue, Sidcup, Kent, DA15 9DL. (01 – 302 – 5427)
CJE Micros	78 Brighton Road, Worthing, W Sussex, BN11 2EN. (0903 – 213361)
Clares Micro Supplies	98 Middlewich Road, Rudheath, Northwich, Cheshire, CW9 7DA. (0606 – 48511)
Computer Concepts	Gaddesden Place, Hemel Hempstead, Herts, HP2 6EX. (0442 – 63933)
Contex Computing	15 Woodlands Close, Cople, Bedford, MK44 3UE. (02303 – 347)
Dudley Micro Services	30 Hadley Close, Netherton, Dudley, DY2 9JX. (0384 – 633142)
EMR Ltd	14 Mount Close, Wickford, Essex, SS11 8HG. (0702 – 335747)
Fairhurst Instruments	Dean Court, Woodford Road, Wilmslow, SK9 2LT. (0625 – 525 – 694)
GEM Electronics	17 Tandragee Road, Portadown, Craigavon, BT62 3BQ.
HopeSoft	Hope Cottage, Winterbourne, Newbury, Berks, RG16 8BB. (0635 – 248472)
HS Software	56, Hendrefolian Avenue, Sketty, Swansea, SA2 7NB. (0792 – 204519)
IFEL	36 Upland Drive, Plymouth, Devon, PL6 6BD. (07555 – 7286)
LTS Ltd	Haydon House, Alcester Road, Studley, Warks, B80 7AN. (0386 – 792617)
MacSoft	36 Alfred Street, Dunstable, Beds. (0582 – 699 – 483)
Maximum	44 Manor Road, Wokingham, Berks, RG11 4AR.
Minerva Systems	69 Sidwell Street, Exeter, EX4 6PH. (0392 – 37756)
Mitre Software	26 Creechurch Lane, London, EC3A 5BA. (01 – 283 – 4646)
Musbury Consultants	8 Fairhill, Helmshore, Rossendale, Lancs, BB4 4JX. (0706 – 216701)
Northern Educational Software	16 Dawson Lane, Bierley, Bradford, BD4 6HN.
Pineapple Software	39 Brownlea Gardens, Seven Kings, Ilford, Essex, IG3 9NL. (01 – 599 – 1476)
RESOURCE	Exeter Road, Doncaster, DN2 4PY. (0302 – 63800/63784)
Solidisk Technology Ltd	17 Sweyne Avenue, Southend-on-Sea, Essex, SS2 6JQ. (0702 – 354674)
Norwich Computer Services	**18 Mile End Road, Norwich, NR4 7QY. (0603 – 507057)**

Archive

The Subscription Magazine for *Archimedes* users

Articles include:

- File transfer on RS423
- Attaching a 5.25" drive
- C.C.'s ROM-Link packages
- Clares' Toolkit module
- Graphics demo programs
- Archie the Music Computer
- Structuring with BASIC V
- Technical notes on BASIC V
- Help with using the ADFS
- Epson Screen Dump
- Using BBC ROM images
- BBC micro as an I/O podule
- Assembly language programming
- Writing relocatable modules
- Using the WIMP environment
- Reviews of software and hardware

Technical Help Service (£8 / year)
A telephone hot-line service for immediate help with your technical problems. Any member can send written enquiries, but for a fast response use the THS!

Members' Discount: 7.5% off software from Computer Concepts, Minerva Systems and Clares Micros Supplies purchased through Norwich Computer Services.

Subscription: 12 issues £12.50 (UK) Europe £18, Middle East £22, America / Africa £25, Elsewhere £27. Technical Help Service £8

* Please send copies of *Archive* magazine for one year starting from
Volume 1 Issue ________

* Please enrol me on the Technical Help Service for one year. (£8)

I enclose a cheque for £ ________

Name: ____________________

Address: ____________________

____

____________________ Postcode: ____________

Norwich Computer Services, 18 Mile End Road, Norwich, NR4 7QY.
Subscription – £10 per year, Technical Help Service – £8 per year